ESCOLA NAVAL

talant de bien faire

Catarina de Sousa e Faro Antunes Pina

Desenvolvimento de um tradutor NMEA para ASPN

Dissertação para obtenção do grau de Mestre em Ciências Militares Navais, na especialidade de Engenharia Naval ramo Armas e Eletrónica

Alfeite
2025

ESCOLA NAVAL

talanti de bien faire

Catarina de Sousa e Faro Antunes Pina

Desenvolvimento de um tradutor NMEA para ASPN

Dissertação para obtenção do grau de Mestre em Ciências Militares Navais, na especialidade de Engenharia Naval ramo Armas e Eletrónica

Orientação de: Vítor Viegas

Co-orientação de: Bruno Damas

O Aluno Mestrando

Assinado por: **Catarina de Sousa e Faro Antunes Pina**
Num. de Identificação: 15468104
Data: 2025.11.07 14:21:00 +0000

Catarina Pina

O Orientador

Assinado por: **Vítor Manuel Rodrigues Viegas**
Num. de Identificação: 10707212
Data: 2025.11.08 16:52:50+00'00'

Vítor Viegas

Alfeite

2025

“Technology, like art, is a soaring exercise of the human imagination.”

Daniel Bell

Dedico este trabalho à minha mãe, pelo apoio incondicional, incentivo constante e por acreditar em mim ao longo de todo este percurso.

Aos meus orientadores, deixo um profundo agradecimento pela orientação, apoio e disponibilidade demonstrados ao longo de todo o desenvolvimento desta dissertação. As suas sugestões e acompanhamento constantes foram determinantes para a concretização deste trabalho.

Um agradecimento especial à minha mãe, pelo apoio incondicional em todas as etapas do meu percurso. A sua força, paciência e dedicação foram sempre o meu maior exemplo e a base que me permitiu chegar até aqui.

Aos meus irmãos, avó, tio e restante família alargada, agradeço pela motivação, pelo incentivo e pelas palavras de confiança, que tantas vezes me deram ânimo nos momentos mais complicados.

Ao meu namorado, agradeço a compreensão, a paciência e o apoio inabalável, que tantas vezes fizeram a diferença o longo desta jornada.

Aos meus melhores amigos, Mateus e Araújo, que caminharam todo este percurso comigo, agradeço a companhia, o entusiasmo e os momentos de descontração que me ajudaram a equilibrar o esforço exigido neste percurso.

Ao Galhanas, que apareceu na etapa final, mas que rapidamente se tornou um pilar essencial, agradeço toda a confiança, o apoio e a forma como trouxe equilíbrio numa fase em que tudo parecia mais difícil.

RESUMO

A presente dissertação aborda a crescente complexidade e diversidade de sistemas de navegação que tornam essencial a criação de soluções capazes de integrar dados provenientes de diferentes fontes de forma fiável e acessível a múltiplas plataformas. Assim, foi desenvolvido um sistema modular destinado à interoperabilidade de dados de Posição, Navegação e Tempo (PNT), com foco na conversão de mensagens segundo o protocolo *National Marine Eletronics Association* (NMEA) para um formato padronizado em *Extensible Markup Language* (XML), seguindo a norma *All Source Positioning and Navigation* (ASPN).

O trabalho desenvolvido incidiu na conceção, implementação e validação de um sistema capaz de receber dados de um recetor GPS, interpretar e validar as mensagens NMEA, convertê-las para XML e redistribuí-las em rede via *User Datagram Protocol* (UDP) *multicast*. O sistema foi desenvolvido segundo uma arquitetura modular, o que permite a integração de componentes específicos em diferentes fases, como um módulo *parser*, tradutor de protocolo, e uma interface gráfica. Esta abordagem garante que existe facilidade na manutenção do sistema bem como o potencial para uma possível expansão futura.

A validação experimental passou por três fases distintas. Inicialmente foram utilizadas mensagens NMEA simuladas em ambiente local, garantindo que o sistema conseguia interpretar os dados recebidos. Uma vez concluída esta fase, integrou-se um módulo u-blox 7M-NEO que fornecia dados *Global Positioning System* (GPS) reais. Os ensaios culminaram num contexto operacional, a bordo de um navio, onde foram testados todos os procedimentos implementados anteriormente. Os resultados obtidos demonstraram que o sistema manteve a consistência ao longo do processo, assegurando a tradução e posterior distribuição em rede.

Os resultados evidenciam que a solução desenvolvida é robusta e adequada para aplicação em cenários reais de interoperabilidade PNT, validando o uso do XML como formato comum para integração de sistemas heterógenos. Este trabalho contribui para

a simplificação da gestão de dados de navegação, promovendo a interoperabilidade entre diferentes equipamentos e protocolos.

Palavras-chave: Interoperabilidade PNT, NMEA, ASPN

ABSTRACT

This dissertation addresses the increasing complexity and diversity of navigation systems, which makes it essential to develop solutions capable of integrating data from different sources in a reliable manner and making it accessible across multiple platforms. Accordingly, a modular system was developed aimed at the interoperability of *Positioning, Navigation, and Timing* (PNT) data, focusing on the conversion of messages following the *National Marine Electronics Association* (NMEA) protocol into a standardized format using *Extensive Markup Language* (XML), in accordance with the *All Source Positioning and Navigation* (ASPN) standard.

The work carried out focused on the design, implementation, and validation of a system capable of receiving data from a GPS receiver, interpreting and validating NMEA messages, converting them to XML, and redistributing them over the network via *User Datagram Protocol* (UDP) multicast. The system was developed using modular architecture, allowing the integration of specific components at different stages, such as a parser module, protocol translator, and a graphical user interface. This approach ensures ease of system maintenance while providing for future expansion.

The experimental validation was conducted in three distinct phases. Initially, simulated NMEA messages were used in a local environment to ensure the system could correctly interpret the received data. Once this phase was completed, a u-blox 7M-NEO module providing real GPS data was integrated. The tests culminated in an operational context, aboard a ship, where all previously implemented procedures were evaluated. The results demonstrated that the system maintained consistency throughout the process, ensuring accurate translation and subsequent network distribution.

The results indicate that the developed solution is robust and suitable for application in real PNT interoperability scenarios, validating the use of XML as a common format for the integration of heterogeneous systems. This work contributes to the simplification of navigation data management, promoting interoperability between different devices and protocols.

Keywords: PNT Interoperability, NMEA, ASPN

DECLARAÇÃO

Declaro que utilizei ferramentas de Inteligência Artificial para correções ortográficas e de semântica do trabalho.

ÍNDICE GERAL

ÍNDICE DE FIGURAS

ÍNDICE DE TABELAS

LISTA DE ABREVIATURAS, SIGLAS E ACRÓNIMOS

ASPN - *All Source Positioning and Navigation*

ENC - *Electronic Navigational Chart*

GPS - *Global Positioning System*

GNSS - *Global Navigation Satellite System*

GUI – *Graphical User Interface*

IEEE - *Institute of Electrical and Electronics Engineers*

IHO - *International Hydrographic Organization*

MMI - *Mixed Mode Interface*

NATO - *North Atlantic Treaty Organization*

NCAP - *Network Capable Application Processor*

NMEA - *National Marine Electronics Association*

NRP - Navio da República Portuguesa

OPC - *Open Platform Communications*

PNT - *Positioning Navigation and Timing*

RFID - *Radio Frequency Identification*

SNMG - *Standing NATO Maritime Group*

SOA - *System Oriented Architecture*

STO - *Science and Technology Organization*

STWS - *Smart Transducer Web Services*

SW - *Semantic World*

TIM - *Transducer Interface Module*

UA - *Unified Architecture*

UDP - *User Datagram Protocol*

W3C - *World Wide Web Consortium*

XML - *Extensible Markup Language*

XSD - *XML Schema Definition*

1. Introdução

A interoperabilidade entre diferentes tipos de sistemas tecnológicos é uma das bases fundamentais na evolução da Engenharia, com principal foco em ambientes operacionais e críticos. A crescente complexidade dos teatros operacionais modernos, juntamente com a participação conjunta de diversas forças aliadas, bem como a disparidade de tipos e fornecedores de equipamentos, evidenciou diversos desafios na comunicação entre sistemas que agora precisam de ser compatíveis entre si.

Na comunidade naval, esta necessidade torna-se particularmente evidente. Numa frota, quer esta seja civil ou militar, são utilizados diversos sistemas de bordo que integram sensores, módulos de comunicação e unidades de processamento. Estes sistemas dependem de protocolos de comunicação padrão para divulgarem a informação que adquirem com celeridade e segurança. A falta de uma norma padronizada e universal pode comprometer a fiabilidade da informação partilhada, podendo incorrer em dificuldades de coordenação e originar riscos operacionais.

O NMEA é um protocolo amplamente utilizados por sistemas *Positioning Navigation and Timing* (PNT). Mesmo com a simplicidade e ampla adoção que o caracteriza, não foi desenhado para colmatar os atuais requisitos impostos por necessidades de interoperabilidade em contextos operacionais heterogéneos. Esta limitação tornou-se evidente em ambientes operacionais da *North Atlantic Treaty Organization* (NATO), onde a interoperabilidade e integração de forças de diferentes países é um requisito base. Foi então criado, dentro da NATO *Science and Technology Organization* (STO), o grupo de trabalho SET-309. Este grupo visa encontrar e estudar soluções que permitam a interoperabilidade de sistemas PNT através da compatibilização das mensagens trocadas entre equipamentos de diferentes fornecedores.

A solução encontrada baseia-se então na criação da norma ASPN, baseada em XML, com capacidade de extensão e maior flexibilidade. Focando-se o grupo de trabalho SET-309 maioritariamente em plataformas terrestres, constatou que, devido à

existência de vários equipamentos que utilizam o protocolo NMEA, seria mais adequado desenvolver um tradutor *plug&play*, que fosse capaz de converter as mensagens NMEA para o formato ASPN em tempo real, evitando assim a necessidade de substituição de equipamentos. Este tipo de solução aparenta adaptar-se também a outros meios, em especial aos meios navais, utilizados pela Marinha Portuguesa.

1.1. Descrição do Problema a Resolver

Os sistemas PNT constituem uma plataforma tecnológica utilizada por uma vasta rede de operações militares e civis. Como a sigla PNT indica, este tipo de plataforma fornece informação geográfica, orientação e sincronização do tempo. Estes sistemas estão presentes em diversos meios, como na navegação marítima, aeronáutica, controlo de tráfego, forças terrestres e mesmo em sistemas de armamento. O PNT é um conceito abrangente, utilizado no quotidiano, que inclui sensores GPS, altímetros, magnetómetros, entre outros, que permitem estimar posição, orientação e tempo com elevada precisão.

Para que os dados produzidos por estes sensores possam ser aproveitados eficazmente, estes devem ser codificados e transmitidos de acordo com protocolos de comunicação específicos. É neste ponto que entra o protocolo NMEA. O protocolo NMEA funciona como uma camada de comunicação entre sistemas PNT e diferentes sistemas embarcados. Desta forma, o protocolo NMEA atua como uma linguagem comum entre sensores e outros equipamentos, focando-se este trabalho sobre os sistemas PNT e os seus utilizadores. Apesar desta vasta utilização, o NMEA enfrenta algumas dificuldades para se adaptar às necessidades de sistemas mais modernos. Com diferentes versões e algumas restrições estruturais torna-se difícil garantir a interoperabilidade podendo comprometer operações de diferentes âmbitos.

Torna-se, portanto, necessário uma abordagem diferente, que mantenha a compatibilidade dos sistemas existentes, e ao mesmo tempo permita a transição para

comunicações mais flexíveis e robustas. É esta a transição que a norma ASPN se propõe a colmatar.

1.2. Motivação

A motivação deste tema de dissertação parte da necessidade de níveis de coordenação e integração tecnológica cada vez mais elevados. Em operações navais, que podem incluir não só navios como também aeronaves e sistemas costeiros, o sucesso depende muito da exatidão e fiabilidade dos dados PNT. Estes dados são a base de diversas operações, desde apenas navegação em segurança até à coordenação tática.

Com o crescente carácter multinacional e interdependente das missões, a exigência para uma partilha de informação segura e compreensível também aumenta. Esta nova realidade impõe desafios, sendo estes percetíveis nas cada vez mais frequentes operações conjuntas da NATO, por exemplo nos *Standing NATO Maritime Groups* (SNMG).

A motivação para esta dissertação surge então da necessidade real de arranjar uma solução para os diversos obstáculos à interoperabilidade entre sistemas PNT nestas forças. A padronização das mensagens é um salto essencial pois permitirá que diferentes sistemas, concebidos por fabricantes distintos, consigam trabalhar entre si.

Como referido anteriormente, a substituição de todos os equipamentos PNT não seria viável, pelo que se optou pelo desenvolvimento de um tradutor *plug&play* que converta automaticamente mensagens PNT no formato NMEA para o formato ASPN. Assim, esta dissertação tem como principal motivação o desenvolvimento do referido tradutor por forma a promover uma continuidade operacional, com custos reduzidos, que leve à interoperabilidade de equipamentos já existentes e em uso com equipamentos novos e futuros.

1.3. Contextualização

Os sistemas PNT são fundamentais para a navegação, pois fornecem dados essenciais como a posição, velocidade e tempo. Estes sistemas são familiares ao meio civil e ao meio militar. Para que diferentes equipamentos possam trocar dados entre si, é necessária a utilização de um protocolo comum. Um protocolo amplamente utilizado é o NMEA 0183. Este protocolo define um conjunto de mensagens que agregam diferentes informações que são transmitidas para posterior processamento.

Atualmente, a interoperabilidade entre sistemas e equipamentos provenientes de diferentes fabricantes e fornecedores é um requisito base no meio operacional. O protocolo NMEA 0183 apresenta várias limitações: é unidirecional, isto é, só permite o envio de um para vários, é pouco flexível, não é legível por humanos e não suporta esquemas de validação.

Com o propósito de facilitar a interoperabilidade entre sistemas foi criada a norma ASPN. Esta norma baseia-se na linguagem de marcação XML, que por sua vez permite maior flexibilidade e compatibilidade entre sistemas, permitindo a troca de informação entre eles, nomeadamente do tipo PNT. Atendendo a que a maioria dos equipamentos marítimos atualmente em uso utiliza mensagens NMEA para transmitir esse tipo de informação, e com vista a poderem integrá-los com equipamentos que utilizem ASPN, a solução passa pelo desenvolvimento de um tradutor que seja capaz de fazer a passagem do protocolo NMEA 0183 para a norma ASPN.

1.4. Objetivos da Dissertação

O objetivo principal do trabalho centra-se no desenvolvimento de um tradutor entre o protocolo NMEA e a norma ASPN, que permita a conversão automática e em tempo real de mensagens PNT, por forma a garantir a interoperabilidade entre diferentes sistemas de navegação. O tradutor deverá ser implementado como um módulo *plug&play* podendo ser facilmente integrado nos sistemas existentes. O principal foco desta parte será garantir a compatibilidade entre equipamentos de diferentes fabricantes.

Por forma a atingir o objetivo principal foram definidos seis objetivos secundários:

- Definição de um algoritmo de divisão das mensagens NMEA por grupos de dados;

- Definição de um algoritmo de tradução da mensagem NMEA para a mensagem ASPN;

- Teste do algoritmo com mensagens NMEA simuladas;

- Teste do algoritmo com mensagens NMEA provenientes de um módulo GPS;

- Adaptação do algoritmo para ser utilizado num sistema embebido, neste caso um *Raspberry Pi 2*, com capacidade *plug&play*;

- Teste do sistema em ambiente real.

Importa referir que o projeto se concentra na conversão baseada no protocolo NMEA 0183, sendo que outros protocolos, como por exemplo outras versões do NMEA, não serão alvo de análise.

1.5. Estrutura da Dissertação

A presente dissertação está dividida em cinco capítulos, abordando assim todas as etapas do projeto. A divisão é a seguinte:

- **Capítulo 2 – Estado da Arte:** Neste capítulo é feita uma análise de diversas plataformas de interoperabilidade como NMEA, IEEE 1451, OPC e ASPN, bem como linguagens de marcação XML. É também realizada uma análise comparativa entre as plataformas enunciadas anteriormente bem como um estudo dos trabalhos já existentes, relacionados com o presente tema;

- **Capítulo 3 – Desenvolvimento:** Este capítulo descreve a conceção e implementação da solução proposta. É abordada a arquitetura do sistema, os módulos que o compõem, e a forma como estes interagem entre si. São ainda detalhados os processos de aquisição, tradução e encaminhamento de dados PNT;
- **Capítulo 4 – Análise de Resultados:** Neste capítulo são apresentados os resultados obtidos com a implementação do sistema. São descritos os testes realizados, analisado o desempenho de diferentes funcionalidades e discutidas as limitações e potencialidades da solução desenvolvida;
- **Capítulo 5 – Conclusões:** O último capítulo sintetiza as principais conclusões do trabalho, destacando contributos para a interoperabilidade em sistemas PNT. São ainda sugeridas melhorias e linhas de investigação futura que poderão potenciar o desenvolvimento da solução apresentada.

2. Estado da Arte

O desenvolvimento de soluções interoperáveis em sistemas PNT exige uma análise dos protocolos e plataformas atualmente disponíveis. A interoperabilidade constitui um requisito essencial em ambientes operacionais cada vez mais complexos, onde a existência de equipamentos de diferentes fabricantes e que operam com diferentes tecnologias é habitual. Assim, torna-se essencial compreender o papel e respetivas normas das principais plataformas de interoperabilidade já estabelecidas no mercado, bem como de normas mais recentes que procurem responder a novas necessidades de segurança e integração. Destas, destacam-se o NMEA 0183, protocolo utilizado no meio marítimo; o IEEE 1451, mais orientado para a padronização de interfaces entre sensores e sistemas de aquisição de dados; e o OPC, utilizado no meio industrial, assegurando comunicação entre diferentes níveis funcionais. A estas juntam-se também linguagens de estruturação de dados, como o XML e, mais recentemente, o ASPN, que procura vingar no meio PNT.

A revisão de leitura apresentada nesta secção tem dois principais objetivos: numa fase inicial pretende proceder a uma análise comparativa entre os diferentes protocolos e plataformas que permitam perceber que plataformas podem suportar a interoperabilidade em sistemas PNT; numa segunda fase, identificar trabalhos publicados que possam ser relevantes na área e que permitam compreender como é que as soluções atualmente existentes têm sido desenvolvidas.

2.1. Plataformas de Interoperabilidade

A interoperabilidade entre sistemas PNT necessita que haja uma garantia de comunicação constante e fidedigna entre equipamentos de diferentes origens. As plataformas de interoperabilidade surgem então como soluções que permitem integrar dados de diferentes sensores e sistemas assegurando, em paralelo, a correta interpretação por parte dos destinatários das mensagens.

Foram selecionadas cinco plataformas e linguagens de estruturação de dados consideradas representativas e relevantes para o estudo:

- NMEA 0183, um protocolo amplamente adotado no meio marítimo, que define um formato de mensagens para dados relacionados com a navegação.

- IEEE 1451, um conjunto de normas que rege a comunicação entre sensores inteligentes;

- OPC, tem origem no setor industrial e garante a interoperabilidade entre sistemas de aquisição e controlo de dados;

- XML, é uma linguagem de marcação bastante versátil e independente da plataforma que representa dados estruturados;

- ASPN, uma norma recente e ainda em desenvolvimento, mais focada no meio operacional.

2.1.1. NMEA 0183

O protocolo NMEA 0183 (NMEA, 2002) é uma norma que foi desenvolvida com o objetivo de padronizar a comunicação e troca de informação entre sistemas PNT marítimos. A transmissão de um equipamento pode ser lida e passada a vários equipamentos, permitindo assim a integração entre sensores e sistemas de diferentes fabricantes.

Este protocolo utiliza mensagens ASCII, estruturadas de forma simples, começando sempre com o carácter "$" e terminando com o carácter "*". Os vários campos, cujo preenchimento é definido para cada tipo de mensagem, são separados por vírgulas. Desta forma, cada mensagem é composta pelos identificadores de início e fim da mensagem, por um identificador do tipo de mensagem e pelos vários campos de dados. A mensagem termina com uma soma de controlo a seguir ao "*", que permite confirmar se a mensagem recebida se encontra correta ou se tem algum erro associado.

O NMEA 0183 suporta mais de 80 tipos distintos de mensagens, sendo que todas são identificáveis por um código próprio. Entre estas estão, por exemplo:

- DBT (*Depth Below Transducer*) - profundidade medida por uma sonda;

- GGA (*Global Positioning System Fix Data*): posição fixa, fornece a latitude, longitude, altitude e qualidade do sinal GPS;

- RMC (*Recommended Minimum Specific GNSS Data*): dados mínimos recomendados para navegação GPS, junta a hora, posição, velocidade e rumo;

- RSA (*Rudder Sensor Angle*): dados de ângulos de leme, transmitidos por um sensor de leme;

- VTG (*Course Over Ground and Ground Speed*): velocidade verdadeira e de superfície, e rumo relativo ao norte magnético e verdadeiro;

Devido à sua simplicidade, é um protocolo de fácil implementação e compreensão, o que ajudou na sua ampla difusão e incorporação. Contudo, a simplicidade também é acompanhada de algumas limitações.

Uma limitação prende-se com a possibilidade de os fabricantes poderem implementar mensagens proprietárias, identificadas pelo prefixo $P. Embora de um ponto de vista esta funcionalidade permita uma maior flexibilidade aos fabricantes, do ponto de vista da interoperabilidade torna-se um problema. Esta liberdade acaba por significar que dois dispositivos que emitam mensagens NMEA proprietárias podem não ser capazes de compreender informação entre si. Além disso, o NMEA 0183 é um protocolo essencialmente unidirecional baseado numa comunicação de um para vários. Isto significa que um emissor transmite para vários recetores, mas não dialogam entre si, o que pode ser uma limitação em cenários mais complexos.

Apesar destas limitações, o NMEA 0183 é um protocolo amplamente adotado, estando presente em inúmeros sistemas civis e militares. A sua ampla adoção tornam-no uma referência na área da navegação e, por esse mesmo motivo, manter a compatibilidade entre novas soluções e este protocolo é um ponto fulcral, sobretudo em sistemas onde coexistam equipamentos de diferentes gerações.

2.1.2. IEEE 1451

A norma IEEE 1451 (Lee, 2009), desenvolvida pelo *Institute of Eletrical and Eletronics Engineers* (IEEE), define um grupo de interfaces normalizadas que têm como objetivo facilitar a comunicação entre transdutores e microcontroladores. O objetivo principal desta norma é promover a interoperabilidade entre sensores e processadores de diferentes fabricantes, reduzindo a necessidade de interfaces proprietárias e facilitando a integração de soluções heterogéneas.

A família de normas IEEE 1451 é composta por várias cláusulas, numeradas por IEEE 1451.x, sendo que o "x" representa o número associado a cada cláusula:

- IEEE 1451.0: constitui a base da família, definindo um conjunto de comandos comuns e, sobretudo, fornecendo os *Transducer Eletronic Data Sheets* (TEDS) (IEEE, 2024). Os TEDS são ficheiros eletrónicos que descrevem as propriedades de um transdutor, tais como tipo, gama, resolução e precisão, tornando-o auto descritivo. Desta forma, qualquer sistema compatível pode reconhecer e configurar de forma independente um sensor, simplificando a sua integração.

- IEEE 1451.1: introduz uma camada de software orientada a objetos para sensores e atuadores, padronizando assim os serviços de rede que lhe estão associados (IEEE, 1999). Esta variante da norma permite que os sensores sejam acedidos remotamente através de uma rede, de forma semelhante a objetos em programação. Esta abordagem trata sensores não apenas como fonte de dados, mas como entidades com métodos e atributos próprios, prontas a serem integradas em arquiteturas mais complexas.

- IEEE 1451.2: define a interface digital entre sensores e processadores de aplicação (IEEE, 1997). Através desta norma é estabelecido um modelo de comunicação independente do fabricante, permitindo que os mesmos sensores possam ser utilizados em diferentes sistemas. Em termos práticos, a norma fornece um protocolo de mensagens entre sensores e processadores de aplicação, garantindo a uniformização da comunicação.

- IEEE 1451.4: define uma *Mixed Mode Interface* (MMI), combinando sinais analógicos e digitais (IEEE, 2004). Esta variante foi criada com o objetivo de garantir a compatibilidade retroativa com sensores analógicos existentes, que continuam a ser utilizados em várias indústrias. Esta variante permite, assim, que um sensor transmita tanto sinais analógicos como dados digitais. Existem dois modos de operação: na Classe 1 o transdutor envia os dados digitais primeiro e só de seguida os analógicos; na Classe 2, os dados digitais e analógicos são enviados em simultâneo em fios paralelos, permitindo maior rapidez.

- IEEE 1451.5: expande o modelo para comunicação sem fios. Esta variante não cria protocolos, mas adota os já estabelecidos, como o Wi-Fi e o Bluetooth (IEEE, 2007). A sua grande mais valia está na sua capacidade de fornecer uma camada padronizada que permite que sensores sem fios sigam e integrem os mesmos princípios de interoperabilidade definidos nas restantes variantes da norma, assegurando que a comunicação *wireless* não compromete a rede.

- IEEE 1451.7: direcionada para sensores baseados em *Radio Frequency Identification* (RFID), esta variante define métodos de comunicação específicos e utilização de TEDS em sistemas que recorrem a etiquetas RFID, sejam estas ativas ou passivas (IEEE, 2010). Assim, facilita a identificação e rastreamento em redes inteligentes de sensores, alargando as possibilidades de aplicação da norma para outras áreas.

O conceito fundamental da norma baseia-se em dois blocos principais (Figura 1), o *Transducer Interface Module* (TIM) e o *Network Capable Application Processor* (NCAP). O TIM é o módulo que está diretamente ligado aos sensores e atuadores, sendo responsável pela aquisição e digitalização dos sinais, bem como do armazenamento dos TEDS. Por sua vez, o NCAP interage com os TIMs integrando-os nas redes de comunicação, funcionando como uma interface entre sensores e sistemas de maior dimensão.

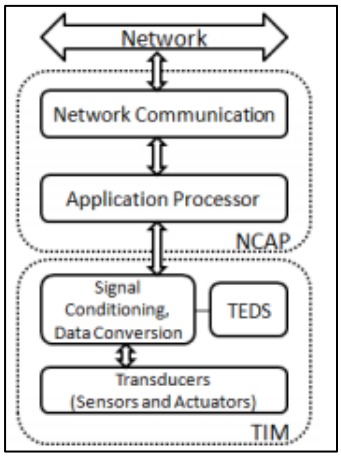

Figura 1 – Modelo de um transdutor inteligente (Junior et al., 2012)

Assim, o IEEE 1451 apresenta uma elevada capacidade de integração com sensores inteligentes e, ao suportar redes com e sem fios, torna-se uma plataforma de interesse num meio onde a integração com diferentes fornecedores de dados é a base para garantir um resultado fiável.

2.1.3. Open Platform Communications

O *Open Platform Communications* (OPC) é uma norma de comunicação utilizada principalmente na automação industrial, tendo como objetivo garantir a interoperabilidade entre dispositivos, sistemas de aquisição e plataformas de controlo de dados de diferentes fabricantes. Antes da sua adoção era comum existirem interfaces proprietárias que dificultavam a integração de sensores. O OPC surgiu como solução deste problema, estabelecendo um modelo de comunicação unificado que permite que diferentes sistemas troquem informação de forma transparente.

O desenvolvimento do OPC começou nos anos 90, de onde resultaram três variantes, cada uma destinada a necessidades específicas. Mais tarde, surgiu uma quarta variante, por forma a colmatar limitações das versões anteriores.

O OPC *Data Access* (DA) foi a primeira especificação desenvolvida e permanece como uma das mais difundidas (OPC Foundation, 2021). Destina-se à comunicação em tempo real, permitindo assim que aplicações de controlo recebam continuamente leituras provenientes de sensores. A sua arquitetura baseia-se no sistema operativo

Windows, o que trouxe alguma simplicidade numa fase inicial, mas acabou por limitar o seu raio de adoção em contextos mais heterogéneos. Apesar desta limitação, o OPC DA consolidou-se como uma referência, servindo como um ponto de partida para normalizar a forma como dados de processo são partilhados em sistemas industriais.

De forma a ultrapassar as limitações do OPC DA relativas ao armazenamento de dados foi criado o OPC *Historical Data Access* (HDA) (OPC Foundation, 2003). Esta variante acrescentou mecanismos para aceder a dados históricos, permitindo consultar e analisar registos anteriores. Esta funcionalidade revelou-se útil em auditorias e estudos de desempenho, bem como no planeamento de manutenções preventivas. Numa vertente técnica, o OPC HDA define métodos específicos para requisitar intervalos e efetuar cálculos de médias ou máximos. Tal capacidade acrescenta valor no meio, pois a evolução temporal dos diversos parâmetros é das ferramentas mais valiosas.

O OPC *Alarms and Events* (AE) foi desenhado para poder lidar com situações em que não chegava conseguir transmitir dados contínuos, era também necessário gerir eventos discretos e alarmes (Emerson, 2016). Assim, esta variante define mecanismos que notificam aplicações quando ocorre algum evento específico, como falhas nos sensores, passagem dos limites de operação ou alertas críticos nos sistemas industriais. Além do aviso emitido, o OPC AE permite estruturar informação adicional, como hora e origem do mesmo. A capacidade de centralizar a gestão dos eventos contribuiu para o aumento da segurança e fiabilidade dos sistemas de monotorização.

Por fim, o OPC *Unified Architecture* (UA) representa uma evolução das versões anteriores, superando as limitações técnicas que as mesmas apresentavam (Automation, 2025). Ao invés da DA, HDA e AE, que estavam ligadas ao ecossistema Windows, o OPC UA foi desenhado para ser independente da plataforma e sistema operativo. A sua arquitetura baseia-se numa *Service Oriented Architecture* (SOA), suportando múltiplos protocolos de transporte, garantido uma possibilidade de escalabilidade em redes mais complexas. (OPC Foundation, 2008)

No âmbito dos sistemas PNT, o OPC pode ser aplicado como camada intermédia de interoperabilidade, facilitando a comunicação entre equipamentos de diferentes fornecedores e que operem com diferentes protocolos de aquisição de dados. Assim, o OPC apresenta-se como uma solução adequada para garantir a interoperabilidade em ambientes complexos e heterógenos, como os que operam sistemas PNT modernos.

2.1.4. Linguagens de Marcação XML

O XML é uma linguagem de marcação utilizada para representar e transferir dados de forma estruturada. Esta linguagem, desenvolvida pelo *World Wide Web Consortium* (W3C) foi projetada para poder ser lida tanto por máquinas como por humanos, permitindo a sua aplicação em múltiplos contextos técnicos. (Bray et al., 2008)

Quanto à sua estrutura, o XML organiza a informação em elementos delimitados por *tags*, que podem conter atributos e valores. Esta estrutura possibilita a representação de dados complexos de forma organizada e facilmente interpretável. Das principais vantagens obtidas por não impor uma estrutura inflexível, é que se torna uma linguagem ideal para incorporar formatos personalizados e adaptados a contextos específicos. Cada ficheiro XML pode ser validado através de *XML Schema Definition* (XSD), que impondo uma estrutura mais restrita, por consequência estabelece também regras adicionais aos tipos de dados e limites de valores, assegura a fiabilidade dos dados trocados entre sistemas. (Gao et al., 2012)

No âmbito da navegação e respetivos sistemas PNT, esta linguagem de marcação pode ser utilizada para diferentes fins. Enquanto ferramenta de interoperabilidade, as suas principais vantagens são a independência da plataforma em que opera, podendo ser lido por praticamente qualquer sistema operativo e linguagem de programação, e a flexibilidade da estrutura de dados. Estas características fazem com que o XML seja útil em ambientes heterogéneos, a nível de sistemas e equipamentos, nos quais seja necessária uma comunicação eficaz e fiável. Além disso, o XML tem a capacidade de suportar metadados, isto é, informação adicional sobre os próprios dados transmitidos.

Esta característica aumenta a fiabilidade de interpretação por parte dos sistemas de destino e facilita a integração em arquiteturas mais complexas.

Esta linguagem de marcação encontra-se aplicada em diversas áreas. No setor marítimo encontra-se implementado nos padrões S-100 da *International Hydrographic Organization* (IHO) (IHO, 2022). Estes padrões definem a estrutura de dados para as *Electronic Navigational Chart* (ENC) bem como dos avisos à navegação e outros produtos hidrográficos. Ao adotar o XML como base, o S-100 permite que haja uma evolução mais fácil do formato dos dados, promovendo assim a interoperabilidade entre diferentes gerações de sistemas de navegação.

Apesar das suas muitas vantagens, o XML apresenta algumas limitações como o aumento da carga de processamento e armazenamento necessária dado que os ficheiros XML tendem a ser mais extensos do que outros formatos. Este fator pode impactar sistemas em tempo real ou ambientes que tenham restrições de largura de banda. Ainda assim, a clareza e extensibilidade, como é possível observar na Figura 2, consolidaram o XML como uma referência no meio dos sistemas de interoperabilidade.

```
<?xml version="1.0"?>
- <birds>
  - <owl id="1201">
      <species>Bubo bubo</species>
      <name>Eagle Owl</name>
      <region>Eurasia</region>
    </owl>
  - <owl id="1202">
      <species>Strix occidentalis</species>
      <name>Spotted Owl</name>
      <region>North America</region>
    </owl>
</birds>
```

Figura 2 – Exemplo de formato de código XML

2.1.5. ASPN

O ASPN é uma norma que foi desenvolvida pela NATO STO com o objetivo de ser utilizada no meio militar pelos países aliados. Este protocolo tem como principal objetivo estabelecer um conjunto padronizado e interoperável de dados oriundos de diversas fontes de dados PNT. O ASPN visa assegurar que, com a crescente complexidade dos teatros operacionais, seja possível coexistirem sistemas de diferentes

gerações, arquiteturas e fabricantes. Nestas circunstâncias a integridade e a fiabilidade da informação PNT é crucial, sobretudo quando existam tentativas de manipulação de sinais (*spoofing*) ou de interferência deliberada (*jamming*).

O ASPN distingue-se pela capacidade de reunir e integrar dados de múltiplas origens em mensagens padronizadas que podem incluir dados em bruto de sensores PNT, estimativas de posição e velocidade e grupo data-hora, entre outros. A norma foi concebida para assegurar que dados PNT sejam partilhados com consistência e precisão, promovendo a confiança operacional. Desta forma, não só garante a interoperabilidade entre sistemas distintos, como também possibilita a verificação da qualidade da informação recebida, reforçando a sua resiliência em cenários contestados.

As principais vantagens do ASPN incluem a interoperabilidade entre equipamentos independentemente do fornecedor e a independência do meio físico, isto é, opera apenas a nível de *software,* não implicando uma mudança de *hardware*, de equipamentos. Em vez de competir com protocolos previamente estabelecidos, como o NMEA 0183, o IEEE 1451 ou o OPC, o ASPN completa-os, atuando como uma camada adicional de interoperabilidade que traduz as mensagens já existentes para o novo formato normalizado. Esta abordagem modular facilita a sua adoção progressiva e reduz os riscos de disrupção de infraestruturas críticas.

Do ponto de vista operacional, o ASPN revela-se particularmente relevante em operações conjuntas multinacionais, nas quais a interoperabilidade entre forças de países aliados constitui um fator determinante no sucesso das missões. A sua utilidade torna-se igualmente evidente em cenários onde o *Global Navigation Satellite System* (GNSS) é degradado ou negado, exigindo a fusão de diferentes tipos de sensores, como sistemas inerciais, para assegurar a continuidade e precisão da navegação.

Assim, o ASPN apresenta uma abordagem centrada na interoperabilidade da gestão de dados PNT em ambientes operacionais. Embora ainda seja uma norma em fase de implementação, o ASPN representa um marco importante na evolução da interoperabilidade em ambiente militar. A sua adoção não só revela resiliência técnica

como também potencia a superioridade estratégica em cenários contestados, sendo adaptável às exigências de missões cada vez mais complexas.

2.1.6. Análise Comparativa das várias soluções

Para ser possível proceder a uma análise comparativa entre as várias plataformas abordadas torna-se necessário definir critérios de comparação. Desta forma, foram definidos como critérios a compatibilidade, a complexidade de implementação e desempenho a nível operacional.

Começando pelo critério da compatibilidade, o protocolo NMEA, mesmo sendo um protocolo amplamente adotado, apresenta limitações a nível de variedade de mensagens padronizadas disponíveis, e na disponibilidade de mensagens proprietárias para diferentes fornecedores. No entanto, devido à sua ampla adoção, é um protocolo com uma presença relevante no meio.

No caso do IEEE 1451, este já foi desenhado com o objetivo de se tornar uma plataforma de interoperabilidade entre sensores e transdutores de diferentes fabricantes. Devido à utilização do TEDS e dos módulos TIM e NCAP, o sistema permite um elevado grau de compatibilidade.

O protocolo OPC UA foi desenvolvido com o objetivo de agrupar e colmatar as limitações dos protocolos OPC já existentes. Assim, é reconhecido como tendo um elevado grau de interoperabilidade, sendo fulcral na integração de redes industriais.

O XML, por ser uma linguagem bastante flexível, permite mensagens bastante personalizadas, tornando-se assim compatível com uma grande variedade de equipamentos e sistemas.

Quanto à norma ASPN, esta destaca-se por ser a mais recente de entre as várias plataformas abordadas. Sendo uma norma baseada em XML, pode antever-se que siga as características do mesmo.

Relativamente à complexidade de implementação, o NMEA, devido à sua simplicidade, é uma solução apelativa. Por sua vez, IEEE 1451 não é tão simples por necessitar de uma estrutura de suporte. No caso do OPC UA, embora seja um protocolo versátil, exige um conhecimento aprofundado do sistema, sendo por isso mais complexo na sua implementação. Por fim, o XML, ao ser bastante versátil, pode exigir outro tipo de ferramentas para validação que tornem a sua implementação mais complicada. Uma vez mais, o ASPN, sendo derivado do XML, deverá apresentar um comportamento similar.

Quanto ao desempenho a nível operacional, torna-se necessário ter conhecimento do contexto em questão. Num contexto em que a largura de banda seja restrita, o NMEA apresenta-se como a solução mais apelativa. Por sua vez, se houver necessidade de maior capacidade de processamento são preferíveis o XML e o OPC UA. O IEEE 1451 apresenta um desempenho intermédio, dependente da infraestrutura e da forma como é implementado.

Plataformas	Compatibilidade	Complexidade de Implementação	Desempenho Operacional
NMEA 0183	Ampla adoção, mas limitado no conjunto de mensagens	Simples, de fácil adoção	Eficiente em contextos de largura de banda restrita
IEEE 1451	Elevado grau de compatibilidade através de TEDS, TIM e NCAP	Mais complexo devido à necessidade de estrutura de suporte	Desempenho adequado, depende da infraestrutura
OPC UA	Elevada interoperabilidade	Complexo, requer conhecimento aprofundado	Bom desempenho, principalmente com maior capacidade de processamento
XML	Muito flexível	Pode exigir ferramentas de validação	Bom desempenho, principalmente com maior capacidade de processamento
ASPN	Flexível	Pode exigir ferramentas de validação	Desempenho previsto similar ao do XML

Tabela 1 – Análise comparativa entre NMEA 0183, IEEE 1451, OPC UA, XML e ASPN

2.2. Trabalhos Relacionados

Esta secção visa abordar e analisar alguns trabalhos relevantes na área da interoperabilidade entre sistemas heterogéneos, particularmente os que apresentam o uso de protocolos anteriormente abordados, como o NMEA, IEEE 1451 e XML.

No estudo *MT-e&R: NMEA Protocol-Assisted High-Accuracy Navigation Algorithm Based on GNSS Error Estimation Using Multitask Learning* (Bao et al., 2022) é apresentado um algoritmo de navegação de precisão denominado MT-e&R. Este algoritmo utiliza mensagens NMEA para recolher dados GNSS, que são posteriormente analisadas. Nesta análise, o algoritmo recorre a multi-task learning para estimar a matriz de covariância, recorrendo aos dados extraídos da posição, velocidade, número de satélites e a qualidade do sinal obtido. Estes dados são suficientes para realizar uma estima coerente. Os resultados obtidos dos vários ensaios mostram que, com a estima realizada, houve reduções entre 35% e 40% do erro de posição.

Em *Service-oriented Sensor Data Interoperability for IEEE 1451 Smart Transducers* (Song & Lee, 2007) os autores centram-se na interoperabilidade em redes de sensores através de uma SOA. Tem como objetivo a criação de uma camada de serviços *web* que exponha os dados dos sensores e sistemas associados. Utiliza a norma 1451 para garantir a estruturação dos dados obtidos. É também desenvolvido um protótipo, o *Smart Transducer Web Services* (STWS), composto por três módulos distintos: um sensor equipado com capacidade de comunicação sem fios (IEEE 1451.5), um módulo NCAP que atua como servidor e expõe as funcionalidades do sensor, e um módulo cliente que deteta os sensores e armazena os dados por eles emitidos.

Em *The XML and Semantic Web Worlds: Technologies, Interoperability and Integration* (Cardoso et al., 2008) os autores abordam a interoperabilidade do XML em comparação com a *Semantic Web* (SW). No caso do XML, este permite estruturar e trocar dados padronizados entre sistemas – interoperabilidade sintática -, no entanto não garante que noutros sistemas esses dados signifiquem o mesmo – interoperabilidade semântica. A SW tenta colmatar essa lacuna através de diversas

tecnologias semânticas. Desta forma, os autores defendem que seria benéfico combinar o XML com a SW, criando assim um sistema mais flexível e interoperável.

Em *ASPN 2023: your community-developed PNT standard* (Schofield et al., 2023) é apresentada uma norma recente para a comunidade científica e industrial. Esta norma surge como resposta à crescente dependência dos serviços PNT baseados em GPS e à necessidade de integrar sensores complementares. O ASPN define um modelo de dados que normaliza a troca de informação entre diferentes sensores e sistemas, garantindo interoperabilidade entre diversas plataformas. Na sua versão atual, contempla a definição do conteúdo de mensagens para 53 sensores distintos. É de destacar o carácter colaborativo desta norma, que conta com a presença de entidades governamentais, académicas e industriais.

3. Metodologia

Este capítulo aborda o processo de conceção e desenvolvimento de uma solução tecnológica que tem como objetivo integrar diferentes formatos de dados provenientes de sistemas PNT. Durante a conceção foi adotada uma abordagem modular permitindo que, à medida que cada componente era desenvolvida, também fosse possível testar e otimizar de forma independente, mas mantendo sempre a coerência da arquitetura global do sistema.

O programa foi concebido para receber dados no formato NMEA 0183 provenientes de diferentes origens, processá-los, traduzi-los e apresentá-los, assegurando a compatibilidade entre diferentes plataformas de diferentes fornecedores. A solução encontrada contempla processos de *parsing* e *deparsing*, conversão para formatos normalizados e, numa fase final, integração com interface gráfica para monotorização.

Nos subcapítulos seguintes é apresentada, numa fase inicial, a visão global da arquitetura que foi desenvolvida, descrevendo-se de forma superficial o seu modo de funcionamento. Posteriormente, é feita uma análise em detalhe dos vários módulos, o seu processo de desenvolvimento e posterior implementação.

3.1. Visão Geral do Sistema

O produto desenvolvido baseia-se numa arquitetura modular que atua como um tradutor entre diferentes protocolos, com especial foco nos dados associados ao PNT, garantindo a interoperabilidade entre sistemas heterogéneos. O principal objetivo é permitir que dados provenientes de diversas fontes, de momento em formato NMEA, sejam processados e reconvertidos para um formato comum padronizado, neste caso o ASPN.

A arquitetura do sistema, representada na Figura 3, foi pensada e desenhada de forma que o fluxo de dados siga uma sequência lógica: numa primeira fase os dados são

captados e passados pelo módulo *parser*. Este módulo vai interpretar e validar a informação recebida segundo o protocolo original, garantindo que apenas os dados corretos e completos avançam para a próxima fase. De seguida, a informação é convertida num formato padronizado, garantindo a preservação de todos os campos de dados relevantes. Após esta conversão, o programa efetua a exportação num formato XML, linguagem na qual o ASPN se baseia, assegurando que os conteúdos dos dados convertidos possam ser lidos por diferentes aplicações e plataformas externas ao programa.

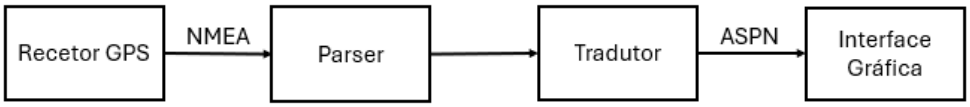

Figura 3 – Arquitetura Geral do Sistema

Numa fase subsequente, entra em ação o módulo de distribuição via UDP *multicast*, responsável por partilhar os dados convertidos com outras máquinas em rede local. Esta opção deve-se à simplicidade e eficiência em cenários onde múltiplos clientes necessitam de receber a mesma informação em tempo real. Ao utilizar *multicast*, o sistema envia apenas uma vez cada mensagem, reduzindo assim a possibilidade de sobrecarga no emissor. Assim, o sistema não só armazena e disponibiliza os dados em ficheiro como também garante a sua passagem imediata a todos os clientes nos diferentes pontos da rede.

Por sua vez, o módulo inversor permite a conversão inversa, permitindo passar de XML para outro protocolo, neste caso NMEA, como ilustrado na Figura 4.

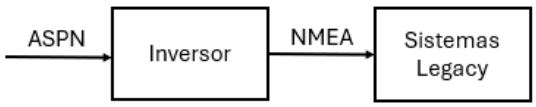

Figura 4 – Arquitetura do Sistema Inversor

Por fim, existe uma interface gráfica (GUI) que possibilita a monitorização e interação com o sistema, em tempo real, de forma intuitiva por parte do utilizador final.

A abordagem modular adotada não só facilita a manutenção e atualização de componentes específicos, como também garante que se for necessário expandir o sistema com novas funcionalidades é mais simples integrar as mesmas sem comprometer o funcionamento global do sistema.

Na criação dos módulos foi utilizada a linguagem de programação *Python*. Esta escolha deveu-se a ser uma linguagem simples e de fácil implementação na plataforma pretendida, *Raspberry Pi*.

3.2. Arquitetura e Principais Componentes

A arquitetura do sistema foi conceptualizada como sendo uma cadeia de processamento, na qual cada módulo desempenha uma função específica, desde a receção de dados até à apresentação dos mesmos através da interface gráfica.

O núcleo é composto por um conjunto de módulos que trabalham de forma conjunta para assegurar o processamento completo dos dados PNT recebidos, desde a sua receção em formato NMEA, até à sua exportação no formato ASPN ou até à sua reconversão.

3.2.1. Módulo Principal

O módulo principal está incorporado no *main.py*, que atua como ponto de entrada do sistema. É este módulo que fica responsável por iniciar a leitura de dados NMEA provenientes do recetor GPS, processá-los através do módulo de *parsing* e por fim, distribuir a informação processada via *socket multicast* UDP. Além disso, grava também os dados convertidos em ficheiros XML, assegurando que os mesmos ficam disponíveis para consultas posteriores.

3.2.2. Módulo Parsing

Este módulo está encarregue de interpretar as mensagens no formato NMEA. Quando recebe uma *string* verifica a conformidade da mesma, valida os *checksums* e extrai apenas os campos selecionados (Longitude, Latitude, Altitude, Horizontal Dilution of Precision (HDOP), Número de satélites em utilização, Velocidade, Grupo Data-Hora,

Métrica representativa do erro de posição, desvios padrão dos erros de longitude, latitude e altitude). Desta forma, o *parser* prepara os dados para serem processados para o formato ASPN.

3.2.3. Módulo Tradutor

Após o *parsing*, este módulo recebe os dados normalizados e gera os ficheiros no formato XML. A estrutura do ficheiro segue o formato do protocolo ASPN, garantindo a compatibilidade com outras aplicações que adotem o mesmo padrão.

3.2.4. Módulo Interface

Este módulo é responsável por implementar a interface gráfica, que permite ao utilizador final visualizar, de forma intuitiva, os dados em formato XML. A visualização dos mesmos é atualizada automaticamente e em tempo real, permitindo ao utilizador uma perspetiva mais realista dos dados processados.

3.2.5. Módulo Inversor

Por fim, este módulo está encarregue de realizar o processo inverso, isto é, converter os dados em formato XML para o formato do protocolo NMEA. Desta forma, fica assegurada a bidirecionalidade do sistema, permitindo não só a exportação de dados em formato padronizado como também a reintegração com sistemas *legacy* ou que apenas aceitem NMEA como entrada. Adiciona também mais uma camada de confirmação.

Esta abordagem modular, além de promover e facilitar a escalabilidade do sistema, permite que qualquer componente seja substituído ou atualizado de forma independente, sem afetar a integridade global do sistema.

3.3. Funcionamento Global do Sistema

O sistema opera segundo uma sequência de operações encadeadas, percetíveis na Figura 5, que permitem receber, processar, converter, enviar e disponibilizar dados

PNT provenientes de diferentes fontes. O principal objetivo é garantir que a informação é tratada de forma uniforme, garantindo a compatibilidade com o formato padronizado ASPN e, quando necessário, a reconversão para protocolos *legacy*.

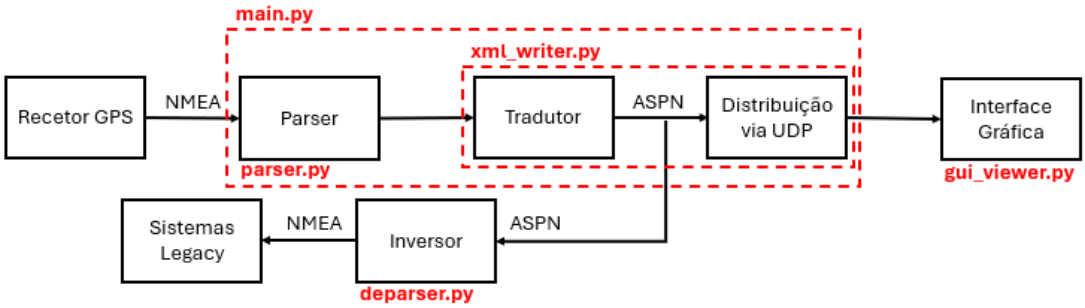

Figura 5 – Funcionamento do Sistema

O processo é iniciado com a receção dos dados NMEA provenientes de um recetor GPS. Estes dados são recebidos no módulo principal (*main.py* - Anexo A) que funciona como um agregador e ponto de coordenação de funções do sistema. A partir daqui as mensagens são encaminhadas para o módulo *parsing* (*parser.py* - Anexo B), onde as *strings* NMEA são analisadas, validadas e organizadas numa estrutura interna de dados normalizada. Durante a validação ocorre a verificação do *checksum* e, se necessário, a exclusão de mensagens corrompidas, garantindo que apenas informação fiável avança para as fases seguintes.

De seguida, os dados normalizados são processados pelo módulo tradutor, (*xml_writer.py* - Anexo C) responsável por gerar a representação no formato ASPN. Em paralelo, são gerados ficheiros XML compatíveis com este padrão. Da mesma forma, o módulo principal assegura que é realizada a distribuição dos dados neste novo formato via o endereço *multicast* UDP 224.0.0.1, possibilitando que múltiplos clientes recebam a informação em simultâneo e em tempo real.

Para uma melhor compreensão dos dados recebidos o módulo interface (*gui_viewer.py* - Anexo E) lê os ficheiros XML de forma automática e apresenta a informação de forma intuitiva ao utilizador final.

Complementarmente, o módulo inversor (*deparser.py* - Anexo F) permite efetuar o processo inverso, reconstruindo as mensagens NMEA a partir dos ficheiros XML em formato normalizado ASPN. Esta funcionalidade é essencial em cenários onde o formato ASPN não seja suportado, podendo ser também utilizada como verificação de *backup*.

De forma geral, o sistema opera como um tradutor entre protocolos, assegurando a integridade da informação bem como a sua interoperabilidade.

3.4. Implementação do Módulo Principal

O ficheiro *main.py* constitui o núcleo do sistema, assumindo a função de coordenador dos processos de aquisição, processamento, conversão e distribuição de dados. Este módulo garante que todos os restantes componentes do sistema são corretamente inicializados e que existe um fluxo de dados contínuo.

Numa fase inicial, é estabelecida a configuração dos parâmetros de funcionamento, como a definição dos endereços e portas para comunicação via *multicast* UDP. É nesta fase que são criados e configurados os *sockets* responsáveis pelo envio dos dados processados aos vários utilizadores.

Após começar a receber sinal do módulo GPS, é invocado o *parser.py* que interpreta as *strings* NMEA recebidas do recetor GPS. Estas linhas de dados são analisadas, validadas e transformadas numa estrutura de dados, já preparada para a conversão. Este processo, através da verificação do *checksum*, garante que apenas as mensagens integras são encaminhadas para as fases seguintes.

De seguida, recorre-se ao *xml_writer.py* que vai gerar os ficheiros XML segundo o formato ASPN. Estes ficheiros são gravados continuamente, criando um registo constante, podendo o mesmo ser utilizado para consultas posteriores.

3.5. Implementação do Módulo Parsing

O módulo *parser.py* é responsável por interpretar e validar as mensagens NMEA que entram no sistema. Este componente foi concebido especificamente para lidar com as particularidades do protocolo NMEA, sendo este último amplamente utilizado em dispositivos GPS e outros sistemas de navegação.

A implementação do *parsing* foi realizada de forma que fossem identificados diferentes tipos de mensagens NMEA e assim fosse possível extrair os campos pretendidos, como a latitude, longitude, hora, e qualidade do sinal.

O processo segue as seguintes etapas:

1. Receção e segmentação da mensagem: cada *string* NMEA é recebida e dividida em campos com base no delimitador padrão, a vírgula;
2. Validação do *checksum*: é realizado o cálculo a partir dos caracteres da frase, e comparado com o valor fornecido no fim da *string*, sendo descartadas quaisquer mensagens em que estes resultados sejam díspares;
3. Extração de campos relevantes: nas mensagens válidas, os campos são segmentados e interpretados segundo o tipo de mensagem a que estão associados;
4. Normalização de dados: a informação extraída é organizada numa estrutura uniforme, independentemente do tipo de mensagem NMEA. Este formato normalizado simplifica o trabalho dos módulos seguintes, garantindo uma maior consistência na conversão para ASPN.

Assim, a função principal deste módulo não se limita à extração de dados, atua também como um filtro de qualidade, assegurando que apenas as informações válidas seguem para o passo seguinte, o tradutor.

3.6. Conversão para ASPN

O módulo xml_writer.py é responsável pela transformação dos dados provenientes do módulo *parsing* no formato ASPN, utilizando XML como linguagem de representação. A sua principal função é garantir que, depois de processados, os dados podem ser interpretados por aplicações externas independentes do protocolo original da mensagem.

É então executado um conjunto de operações sequenciais:

1. Receção dos dados normalizados: os dados passados pelo módulo de *parsing* são transmitidos ao *xml_writer.py* com uma estrutura consistente, que assegura que todos os campos relevantes estão presentes;
2. Mapeamento: cada campo da estrutura anterior é associado ao elemento XML correspondente. Esta correspondência garante a compatibilidade com outros sistemas que utilizem o mesmo padrão;
3. Criação do documento XML: o módulo cria um ficheiro XML, mantendo a formatação de forma a facilitar a leitura, tanto para máquinas como para o utilizador;
4. Ficheiro persistente: o ficheiro XML é guardado no sistema, ficando disponível para análises e consultas futuras.

Além de ser responsável pela conversão e escrita dos ficheiros XML, este módulo é também projetado para ser integrável com o sistema de distribuição de dados em tempo real. Desta forma, o ficheiro XML gerado é armazenado e enviado via *multicast* UDP em simultâneo, garantindo a disponibilidade imediata da informação.

3.7. Interface Gráfica

O módulo relativo à interface gráfica, *gui_viewer.py*, tem como objetivo proporcionar ao leitor uma visualização intuitiva e disponibilizada em tempo real dos dados que foram processados pelo sistema. Esta interface, garante um ponto de

interação mais próximo entre o operador do sistema e a informação recolhida de gerada.

O funcionamento desde módulo assenta nos seguintes princípios:

1. Leitura dos ficheiros XML: o ficheiro *gui_viewer.py* monitoriza continuamente a pasta de *output* dos ficheiros XML e, ao detetar novos ficheiros, expõe os mesmos;
2. Atualização em tempo real: à medida que os dados vão sendo processados e exportados, a interface é atualizada automaticamente, permitindo ao utilizador acompanhar a evolução da informação;
3. Apresentação estruturada da informação: os dados processados são organizados de forma clara. São destacados campos específicos, como a latitude, longitude, grupo data-hora, que facilitam a leitura por parte do utilizador;
4. Interface intuitiva: foi criado um *design* simples, que garante não ser necessário qualquer tipo de formação para se interpretar os dados expostos.

A introdução desta GUI possibilita não só a verificação visual dos dados, mas também a deteção de anomalias ou possíveis falhas de transmissão. Assim, o *gui_viewer.py* contribui para uma solução mais robusta, complementando a vertente técnica da mesma com um elemento mais acessível.

3.8. Implementação do Módulo Inversor

O módulo *deparser.py* garante a implementação da funcionalidade inversa ao processo de *parsing*, permitindo a conversão dos dados no formato ASPN de volta para o protocolo NMEA.

Este módulo segue as seguintes etapas:

1. Leitura do ficheiro XML: o módulo inicia a sua função ao carregar o ficheiro

XML criado pelo *xml_writer.py* aquando da sua conversão;

2. Extração de dados relevantes: tal como foi realizado na conversão inicial, são identificados os campos relevantes e necessários para a construção de uma *string* NMEA;
3. Formatação para NMEA: os dados extraídos do ficheiro XML são organizados segundo a sintaxe própria de cada tipo de mensagem NMEA. É também calculado o *checksum*, que deverá ser igual ao inicial;
4. Validação final: antes de a mensagem ser enviada é feita uma verificação de conformidade, garantindo assim que a mesma será aceite por sistemas externos;

Este módulo desempenha um papel duplo no sistema. Num primeiro ponto de vista, permite a integração com sistemas *legacy* ou que apenas consigam processar dados NMEA. Num segundo, possibilita a existência de mais um passo de verificação, ao ser possível verificar se os dados mantêm a sua integridade quando reconvertidos ao seu formato original.

3.9. Execução e Integração no Raspberry Pi

Por forma a ser possível validar o sistema desenvolvido em ambiente real, todos os módulos foram integrados e executados numa plataforma *Raspberry Pi 2*. Esta escolha foi feita com base na versatilidade da plataforma e facilidade de integração de elementos periféricos externos. Assim, esta opção permite a replicação do sistema em ambiente controlado.

A execução do sistema no *Raspberry Pi* segue as seguintes etapas:

1. Configuração do ambiente: é necessário proceder à instalação de diversos pacotes, configurações de permissões e preparação da rede por forma a proceder ao envio de mensagens via UDP *multicast*;
2. Ligação ao recetor GPS: o *Raspberry Pi* estabelece a comunicação direta com

o dispositivo GPS, a partir do qual são recolhidos os dados no formato NMEA;

3. Execução do módulo principal: o programa é iniciado, o que desencadeia uma cadeia composta pelos módulos principal, *parsing*, tradutor e distribuição via endereço *multicast*;
4. Interação com a GUI: permite a monotorização em tempo real dos dados convertidos através de uma interface gráfica;

A utilização do *Raspberry Pi* demonstra a portabilidade da solução desenvolvida, demonstrando que o sistema pode ser executado em dispositivos com recursos computacionais limitados sem comprometer a fiabilidade e viabilidade do processo. Da mesma forma, este modelo de implementação constitui um avanço para a sua aplicação em contexto real, seja em plataformas terrestres móveis, navios ou outros sistemas que impliquem monotorização remota.

Além destas etapas, foi também configurado o arranque automático do sistema no *Raspberry Pi*, garantindo que o processo é iniciado sem necessidade de intervenção do utilizador. Para isso, foi criado um serviço dedicado no *systemd*, documentado no Anexo G, responsável por executar o script principal, main.py, e assegurar a sua reinicialização em caso de falha. Assim, o sistema encontra-se preparado para funcionar de forma contínua em cenários reais de operação.

4. Resultados Experimentais

A análise dos resultados experimentais constitui uma etapa essencial na validação do sistema desenvolvido, permitindo verificar se os objetivos definidos foram cumpridos e em que medida a solução proposta consegue assegurar a interoperabilidade dos dados PNT através de uma abordagem baseada em XML. Mais do que uma simples verificação, esta fase representa a transição entre a conceção teórica do sistema e a sua utilização prática em cenários reais.

O propósito principal da análise é comprovar que o sistema consegue receber dados PNT em formato NMEA, traduzi-los para XML e redistribuí-los em rede, assegurando a consistência e fiabilidade durante todo o processo. É através desta validação que se consegue verificar a adequação do sistema para uma futura integração em ambientes operacionais.

4.1. Metodologia dos Testes

Os testes executados ao sistema foram realizados de forma incremental, reduzindo a complexidade inicial e permitindo uma validação progressiva de cada componente do sistema. Esta abordagem permitiu garantir que, em cada etapa, os resultados obtidos correspondiam às expectativas antes de avançar para outro mais complexo, reduzindo a probabilidade da ocorrência de erros acumulados.

Numa primeira fase recorreu-se a um ambiente estritamente local, um computador pessoal, como plataforma de desenvolvimento e testes. Nesta etapa foram utilizadas mensagens NMEA simuladas, de modo a confirmar que o processo de tradução para XML funcionava corretamente e que o sistema conseguia gerar ficheiros de saída válidos (*fake_gps_input.py* - Anexo H). Esta etapa foi essencial para consolidar a base funcional do sistema antes de introduzir hardware real.

Numa segunda etapa integrou-se o *hardware*, com a utilização de um módulo GPS da marca u-blox, modelo NEO-7M. O módulo foi configurado através da ferramenta *u-center* para transmitir apenas um subconjunto específico de mensagens: GGA (posição

fixa), GST (estatísticas de precisão), GLL (latitude e longitude) e RMC (dados mínimos recomendados para navegação GPS). A restrição a este conjunto de mensagens teve como objetivo focar apenas as mensagens mais relevantes para a caracterização da posição, tempo e qualidade da navegação, bem como simplificar a análise de resultados. Esta fase permitiu validar a capacidade do sistema em lidar com dados reais provenientes de um recetor GNSS.

Numa fase final, os ensaios abrangeram um contexto mais realista, recorrendo ao equipamento GPS (Furuno GP-32) instalado a bordo do Navio da República Portuguesa (NRP) Zarco. Este equipamento encontra-se encastrado pelo que não foi possível aceder diretamente à sua porta de série. No entanto, e por forma a permitir a transmissão dos seus dados para outros equipamentos, o Furuno GP-32 encontra-se ligado a um sistema de gestão, *Device Master* RTS8 DB9, que recebe e retransmite esses dados através de protocolo TCP/IP. Por forma a obter os dados necessários, foi utilizado um *switch* para fazer as ligações necessárias entre o RTS8, o *Raspberry Pi* e o computador portátil (cliente), ilustradas na Figura 6 e conforme esquema da Figura 7. Foi necessário identificar, através da ferramenta *Wireshark*, a porta de transmissão TCP utilizada pelo equipamento e redirecionar o fluxo de dados para o *Raspberry Pi*, que assumiu o papel de servidor responsável pela tradução e redistribuição via UDP *multicast*. Nesta configuração, o computador pessoal funcionou como cliente, recebendo as mensagens processadas e validando a disseminação de informação em rede.

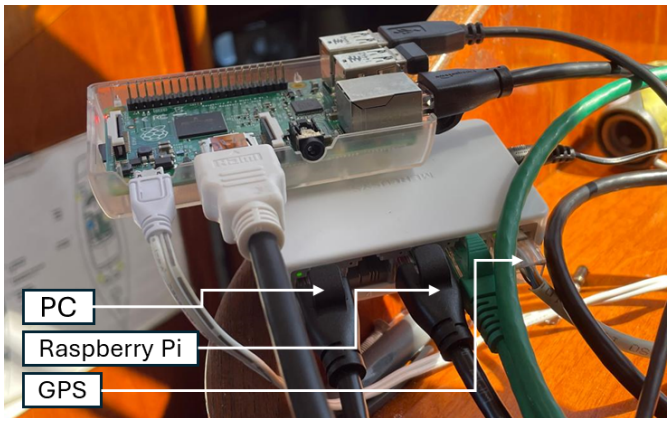

Figura 6 – Raspberry Pi montado a bordo do NRP Zarco

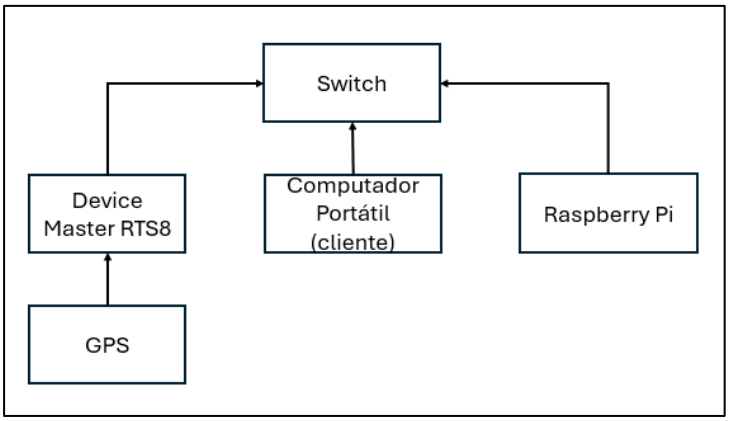

Figura 7 – Esquema de ligação dos equipamentos

Em todas as fases de ensaio, foi alocada especial atenção na validação funcional do sistema garantindo assim a receção de mensagens NMEA, tradução para ASPN, geração de ficheiro de saída e distribuição via *multicast*. Esta metodologia permitiu que o sistema fosse avaliado em condições de complexidade crescente, refletindo cenários de desenvolvimento e cenários mais realistas.

4.2. Resultados Obtidos

Os testes realizados permitiram avaliar de forma gradual a robustez e fiabilidade do sistema desenvolvido, abrangendo desde mensagens simuladas em ambiente local até dados reais obtidos de um recetor instalado a bordo de um navio. Esta abordagem incremental foi fundamental para validar cada componente do sistema antes de proceder a integrações mais complexas.

Numa primeira fase foram utilizadas mensagens NMEA simuladas em *loop*, abrangendo uma mensagem de cada tipo considerado relevante para a análise de posição e navegação: GGA, GST, GLL e RMC. Esta fase permitiu confirmar que o processo de tradução para ASPN estava a ser executado corretamente, garantindo que cada campo das mensagens era preservado e convertido para a estrutura XML sem perdas de informação ou inconsistências, como representado na Figura 8. Nos ensaios iniciais, a

consistência dos dados foi mantida ao longo dos vários ciclos de teste, com as atualizações a ocorrerem de forma regular a cada segundo, de acordo com a Figura 9.

```
<?xml version='1.0' encoding='utf-8'?>
<GPSData>
<latitude>49.274166666666666</latitude>
<longitude>-123.18533333333333</longitude>
<altitude>545.4</altitude>
<hdop>0.9</hdop>
<satellites>8</satellites>
<speed>22.4</speed>
<timestamp>1994-03-23 12:35:20+00:00</timestamp>
<rms>1.2</rms>
<std_dev_lat>0.5</std_dev_lat>
<std_dev_long>0.4</std_dev_long>
<std_dev_alt>1.0</std_dev_alt>
</GPSData>
```

Figura 8 – Ficheiro ASPN resultante das mensagens NMEA simuladas

```
[GGA] Satélites: 08, HDOP: 0.9, Altitude: 545.4 m
[RMC] Hora: 1994-03-23 12:35:20+00:00, Velocidade: 22.4 nós
[GLL] Coordenadas: 49.274166666666666 N, -123.18533333333333 W
[GST] RMS: 1.2, Erro Lat: 0.5, Lon: 0.4, Alt: 1.0
[GGA] Satélites: 08, HDOP: 0.9, Altitude: 545.4 m
[RMC] Hora: 1994-03-23 12:35:20+00:00, Velocidade: 22.4 nós
[GLL] Coordenadas: 49.274166666666666 N, -123.18533333333333 W
[GST] RMS: 1.2, Erro Lat: 0.5, Lon: 0.4, Alt: 1.0
[GGA] Satélites: 08, HDOP: 0.9, Altitude: 545.4 m
[RMC] Hora: 1994-03-23 12:35:20+00:00, Velocidade: 22.4 nós
[GLL] Coordenadas: 49.274166666666666 N, -123.18533333333333 W
[GST] RMS: 1.2, Erro Lat: 0.5, Lon: 0.4, Alt: 1.0
```

Figura 9 – Dados extraídos pelo parser das mensagens NMEA simuladas

Na fase seguinte foi feita a integração do módulo u-blox NEO-7M, que permitiu trabalhar com dados reais de GNSS. Após parametrização do módulo, o sistema demonstrou capacidade de lidar com o fluxo de dados reais, mantendo a tradução correta para XML, bem como a geração de ficheiros de saída consistentes (Figura 10) e a disponibilização dos dados para análise e redistribuição. À semelhança do que aconteceu no ambiente simulado, a cada segundo o sistema recebia e processava dados.

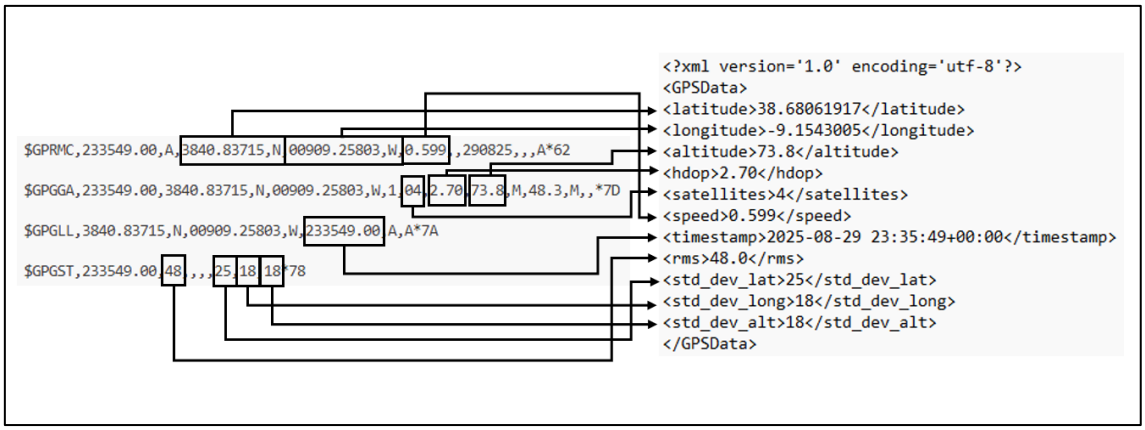

Figura 10 - Correspondência entre campos da mensagem NMEA e campos da mensagem ASPN

A validação foi realizada através da reconversão das mensagens XML de volta para NMEA e pela visualização dos dados na interface gráfica (Figura 11), confirmando que o sistema preservava a fidelidade da informação ao longo do processo.

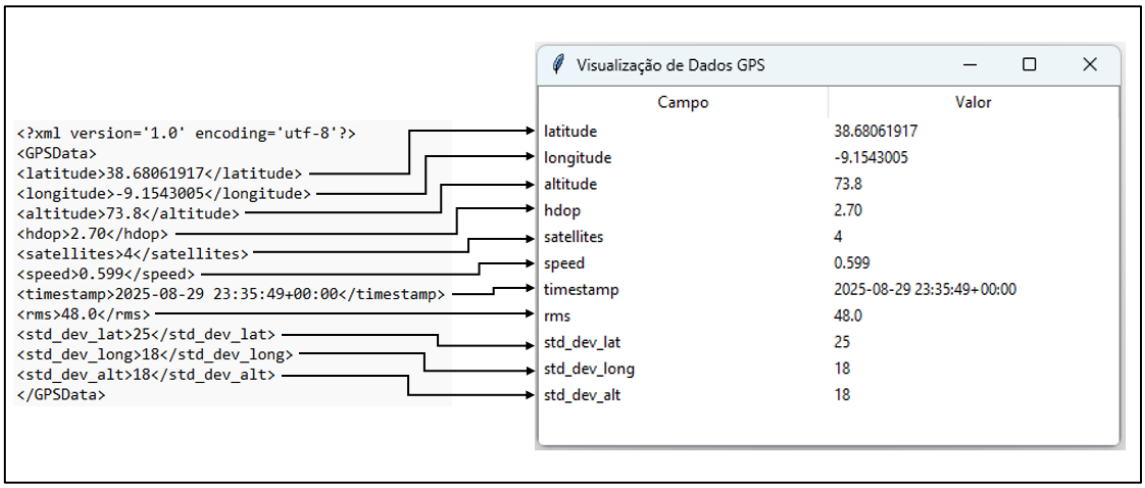

Figura 11 - Correspondência entre os campos da mensagem ASPN e os campos da Interface Gráfica

Nos ensaios da fase final em ambiente real, recorrendo ao equipamento GPS instalado a bordo de um navio, o sistema voltou a evidenciar consistência e fiabilidade. A ligação via TCP ao *Raspberry Pi* garantiu a receção contínua das mensagens, que foram traduzidas e redistribuídas via UDP *multicast (*Figura 12*)*. O sistema manteve a sua performance e consistência, não sendo detetadas interferências ou perdas relevantes.

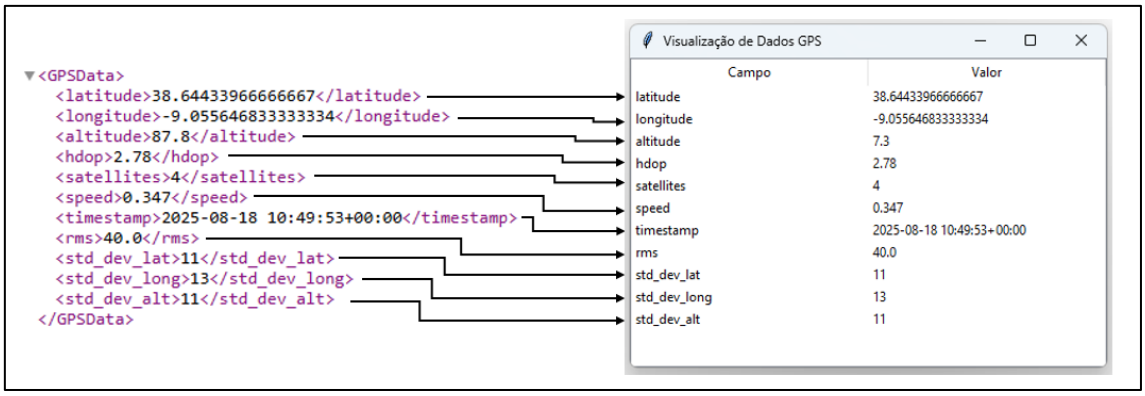

Figura 12 - Correspondência entre campos da mensagem ASPN e os campos da Interface Gráfica

O uso do endereço *multicast* 224.0.0.1 permitiu a distribuição de dados para clientes em rede local (Figura 13), demonstrando que a abordagem adotada é viável para cenários operacionais onde múltiplos sistemas podem necessitar de receber os mesmos dados simultaneamente ou com latências curtas. O *Raspberry Pi* apresentou um desempenho satisfatório, notando-se uma ligeira degradação na velocidade de certos processos paralelos (abertura de ficheiros e movimento do cursor) atribuídas à sua capacidade de processamento limitado, mas sem nunca comprometer a operação global do sistema.

```
[Multicast Server] À escuta em 224.0.0.1:5000
[Multicast Server] De ('192.168.1.86', 65197): [RMC] Hora: 2025-08-28 00:55:19+00:00, Velocidade: 0.113 nós

[Multicast Server] De ('192.168.1.86', 65197): [GGA] Satélites: 06, HDOP: 2.42, Altitude: 54.6 m

[Multicast Server] De ('192.168.1.86', 65197): [GLL] Coordenadas: 38.68085583333333 N, -9.154478833333334 W

[Multicast Server] De ('192.168.1.86', 65197): [GST] RMS: 41.0, Erro Lat: 13, Lon: 11, Alt: 14

[Multicast Server] De ('192.168.1.86', 65197): [RMC] Hora: 2025-08-28 00:55:20+00:00, Velocidade: 0.099 nós

[Multicast Server] De ('192.168.1.86', 65197): [GGA] Satélites: 06, HDOP: 2.42, Altitude: 54.7 m

[Multicast Server] De ('192.168.1.86', 65197): [GLL] Coordenadas: 38.68085466666667 N, -9.1544775 W

[Multicast Server] De ('192.168.1.86', 65197): [GST] RMS: 40.0, Erro Lat: 13, Lon: 11, Alt: 14
```

Figura 13 – Mensagens recebidas pelo cliente

Caso o checksum de alguma das mensagens NMEA não esteja correto, o sistema apresenta a mensagem “[Erro] Mensagem inválida”. Esta mensagem foi obtida na primeira fase de testes, de forma propositada, não se tendo verificado noutras fases, como ilustrado na Figura 14. Adicionalmente, um dos campos da mensagem GGA serve

para indicar se o equipamento já obteve um *fix* GPS, apresentando neste caso o valor de “1”, e caso contrário o valor de “0”, sendo que nessa situação o sistema devolve a mensagem “[GGA] sem fix.”. Esta mensagem surgiu apenas na segunda fase de testes quando o módulo u-blox não conseguia um *fix*. Não foi registada nenhuma destas mensagens durante a fase final de testes.

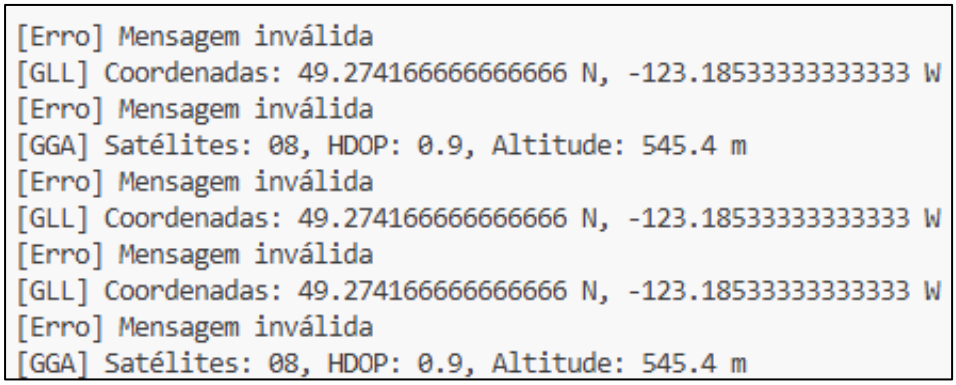

Figura 14 – Mensagem de erro “Mensagem inválida”

No conjunto das fases de testes, os resultados obtidos evidenciaram que o sistema desenvolvido é capaz de receber dados PNT em formato NMEA, traduzir para ASPN, gerar ficheiros de saída válidos e redistribuir os dados via rede de forma consistente e fiável. Estes resultados confirmam que o sistema é adequado para aplicação em ambientes operacionais, demonstrando a sua robustez, fidelidade na tradução de dados e capacidade de distribuição eficiente.

4.3. Discussão de Resultados

A análise dos resultados obtidos permite confirmar que o sistema desenvolvido cumpriu os objetivos fundamentais definidos na fase inicial, demonstrando capacidade para processar mensagens NMEA, convertê-las para XML e redistribuí-las em rede.

Do ponto de vista da robustez e fiabilidade, o sistema demonstrou ser bastante estável em todos os cenários de teste. Tanto no ambiente simulado como com dados reais provenientes do módulo GPS e do equipamento de bordo, o processamento das

mensagens manteve-se consistente, sem perdas ou corrupção de dados. A ocorrência de mensagens de erro antes da obtenção de *fix* GNSS foi um evento considerado normal e expectável, e inerente ao funcionamento do recetor utilizado. O único constrangimento detetado foi a lentidão ocasional do *Raspberry Pi*, sendo a mesma associada às suas limitações de capacidade de processamento. Assim, pode surgir a necessidade de recorrer a um *hardware* mais robusto.

Relativamente à interoperabilidade e à consistência dos dados, a tradução das mensagens NMEA para ASPN revelou-se eficaz, preservando todos os campos e assegurando a fidelidade da informação original. A reconversão das mensagens e a utilização da interface gráfica para verificação confirmaram a ausência de perdas de informação. Este resultado é particularmente relevante pois reforça a viabilidade da possibilidade de adoção de XML como distribuidor de dados PNT.

Foi também possível identificar algumas limitações nos testes realizados. A primeira prende-se com a ausência de testes com múltiplos clientes em simultâneo. Uma vez que apenas foram realizados testes com um cliente (computador portátil), não foi possível observar o impacto em termos desempenho e consumo de memória que esta alteração traria. Este aspeto constitui uma vertente importante para trabalhos futuros dado que em cenários reais será normal a necessidade de partilhar os mesmos dados com vários sistemas em simultâneo.

Quanto ao emprego operacional da solução, os ensaios a bordo do NRP Zarco mostraram que o sistema é integrável num contexto real, processando e redistribuindo dados GPS sem comprometer a sua integridade. No entanto, a solução apresentada ainda não possui maturidade computacional suficiente para ser implementada em contextos mais complexos, onde já não haja tolerância a falhas ou ligeiros acertos. Para se aproximar desse patamar, seria necessário:

- Alargar o conjunto de mensagens NMEA suportadas;

- Realizar testes com múltiplos clientes;

- Garantir compatibilidade com diferentes infraestruturas de rede;

- Explorar mecanismos de monitorização.

A decisão de restringir o sistema a um conjunto mais reduzido de mensagens NMEA revelou-se adequada na fase de validação, pois permitiu reduzir a complexidade e cingir o foco nos módulos de tradução e de distribuição dos dados. No entanto, a mesma restrição limita o alcance da solução. A decisão de adotar o XML como formato de distribuição traz alguma redundância em termos de tamanho à mensagem face ao NMEA, no entanto é facilmente justificada pois permite uma flexibilidade e capacidade de integração que não existe noutras linguagens.

Concluindo, a análise de resultados experimentais permitiu validar a eficácia do sistema desenvolvido, demonstrando que a abordagem baseada em XML garante a correta tradução e redistribuição de dados PNT em diferentes contextos operacionais. Os testes realizados, desde mensagens simuladas até aos dados obtidos a bordo de um navio, demonstraram a robustez e fiabilidade da solução proposta. Esta avaliação fornece uma base sólida para a reflexão sobre aplicabilidade prática do sistema e serve de ponto de partida para a consideração de melhorias e outros desenvolvimentos futuros.

5. Conclusões

5.1. Conclusão

O trabalho desenvolvido teve como objetivo principal a conceção e implementação de um sistema modular que fosse capaz de assegurar a interoperabilidade de dados PNT, através da tradução de mensagens em formato NMEA para XML, numa estrutura prevista pela norma ASPN. Em simultâneo, era necessário garantir que o sistema efetuava a conversão de dados preservando a integridade dos mesmos.

O desenvolvimento seguiu uma abordagem faseada, que se revelou determinante na validação de cada componente do sistema. Numa primeira fase, a utilização de mensagens NMEA simuladas permitiu implementar o processo de *parsing* e tradução, garantindo que o sistema mantinha consistência nos vários ciclos de teste. Na segunda fase, foi integrado o módulo u-blox 7M-NEO que possibilitou a transição para a utilização de dados reais, permitindo confirmar que o sistema conseguia lidar com um fluxo contínuo de dados, sem comprometer a informação. Por último, foram realizados ensaios num cenário real, a bordo de um navio da Marinha Portuguesa, constituíram o teste final, expondo o sistema a condições de operação reais.

Importa destacar a adoção da distribuição via UDP *multicast*, que se espera vir a ser uma solução eficaz para a partilha de dados com múltiplos clientes em rede local. A interface gráfica complementou o sistema, tornando-o mais acessível, funcional e facilmente monitorizado.

De forma geral, os resultados obtidos vão de encontro aos objetivos definidos inicialmente. O sistema é capaz de receber dados em formato NMEA, traduzi-los para XML, gerar ficheiros válidos e redistribuir a informação consistentemente via UDP *multicast*. A arquitetura modular mostrou-se útil, não apenas pela facilidade de manutenção do sistema como também pela flexibilidade que permite para realizar algumas alterações.

Assim, conclui-se que o projeto é capaz de operar tanto em cenários de teste como em cenários reais. Além de validar a aplicabilidade técnica da solução, reforça também a importância da interoperabilidade e da padronização de dados no domínio do PNT.

5.2. Trabalhos Futuros

O sistema desenvolvido apresentou resultados positivos, no entanto há espaço para melhorar a solução inicialmente apresentada. Foram selecionadas três vertentes a melhorar: ao nível da distribuição dos dados em rede, software e hardware.

Relativamente à distribuição de dados, é de destacar a necessidade de realização de testes com múltiplos clientes em simultâneo. Durante o projeto, a distribuição de dados via UDP *multicast* foi validada apenas com um cliente. Considera-se então essencial avaliar o comportamento do sistema em redes mais complexas, com vários dispositivos a acederem em paralelo ao mesmo fluxo de dados.

O sistema foi testado com um número muito restrito de mensagens NMEA, apenas quatro. Assim sendo, uma vertente a desenvolver será ampliar o conjunto mensagens que o *parser* consegue suportar. Ao alargar esta seleção, a compatibilidade com outros equipamentos também aumenta. Esta evolução mexe na vertente de software do sistema, garantindo assim que novas mensagens NMEA também são processadas no módulo de *parsing*, por forma a que a tradução para XML continue a ser feita de forma consistente.

O *Raspberry Pi* revelou ter sido escolha adequada, mas, como referido na discussão de resultados, a utilização de dispositivos com maior capacidade de processamento deve ser uma opção a ponderar. A escolha de um modelo mais recente do *Raspberry Pi* ou de outro equipamento deve ser considerada se se proporcionar um cenário no qual o volume de dados é mais elevado. Desta forma, um próximo passo passaria sempre por equacionar melhorar o *hardware* utilizado.

Referências Bibliográficas

Automation, S. I. (2025). *OPC UA Expertise*. https://industrial.softing.com/expertise/opc-ua.html

Bao, L., Luo, H., Gao, X., Ning, B., & Zhao, F. (2022). *MT-e&R: NMEA Protocol-Assisted High-Accuracy Navigation Algorithm Based on GNSS Error Estimation Using Multitask Learning*.

Bray, T., Paoli, J., Sperberg-McQueen, C., Maler, E., & Yergeau, F. (2008). *Extensive Markup Language (XML) 1.0*. https://www.w3.org/TR/2008/REC-xml-20081126/

Cardoso, J., Hepp, M., & Lytras, M. (2008). *The XML and Semantic Web Worlds: Technologies, Interoprability and Integration*.

Emerson. (2016). *OPC Alarms and Events Overview*.

Gao, S., Sperberg-McQueen, C., Thompson, H., & Keio. (2012). *W3C XML Schema Definition Language (XSD) 1.1*. https://www.w3.org/TR/xmlschema11-1/

IEEE. (1997). *IEEE 1451.2*. https://standards.ieee.org/ieee/1451.2/2165/

IEEE. (1999). *IEEE 1451.1*. https://standards.ieee.org/ieee/1451.1/2164/

IEEE. (2004). *IEEE 1451.4*. https://standards.ieee.org/ieee/1451.4/2168/

IEEE. (2007). *IEEE 1541.5*. https://standards.ieee.org/ieee/1451.5/3299/

IEEE. (2010). *IEEE 1451.7*. https://standards.ieee.org/ieee/1451.7/4582/

IEEE. (2024). *IEEE 1451.0*. https://standards.ieee.org/ieee/1451.0/11001/

IHO. (2022). *S-100 Universal Hydrographic Data Model*. https://iho.int/en/s-100-universal-hydrographic-data-model

Junior, A., Pfleger, S., & Andrade, R. (2012). *IEEE 1451 com TIM e NCAP unificados*. https://lisha.ufsc.br/teaching/dos/ine5424-2012-1/work/g4/

Lee, K. (2009). *IEEE 1451 and IEEE 1588 Standards*.

NMEA. (2002). *NMEA 1083—Standards for Interfacing Marine Electronic Devices*.

OPC Foundation. (2003). *OPC Historical Data Access Specifications*. https://opcfoundation.org/developer-tools/specifications-classic/historical-data-access/

OPC Foundation. (2008). *Unified Architecture—Landingpage*. https://opcfoundation.org/about/opc-technologies/opc-ua/

OPC Foundation. (2021). *OPC DA 2.05 Specification*.

Schofield, A., Bentz, M., Raquet, J., & Kauffman, K. (2023). *ASPN 2023: Your community developed PNT standard*. https://www.spiedigitallibrary.org/conference-proceedings-of-spie/12544/125440M/ASPN-2023-your-community-developed-PNT-standard/10.1117/12.2664201.short

Song, E., & Lee, K. (2007). *Service-oriented Sensor Data Interoperability for IEEE 1451 Smart Transducters*.

Anexos

Anexo A – main.py

```
import serial
import socket
from parser import NMEAParser
from xml_writer import XMLWriter

#porta_serie = "COM3" #para PC com u-blox

#porta_serie = "/dev/ttyUSB0" #para Raspberry com u-blox
#baud_rate = 9600             #para Raspberry com u-blox

#FURUNO_IP = "192.168.250.250" #para Raspberry com Furuno
#FURUNO_PORT = 4606           #para Raspberry com Furuno


parser = NMEAParser()
writer = XMLWriter()

# endereço multicast
UDP_IP = "224.0.0.1"   #grupo multicast
UDP_PORT = 5000
udp_socket = socket.socket(socket.AF_INET, socket.SOCK_DGRAM,
socket.IPPROTO_UDP)

# TTL -> alcance da multicast, 1 = só rede local)
udp_socket.setsockopt(socket.IPPROTO_IP, socket.IP_MULTICAST_TTL, 1)

try:
    gps = serial.Serial(porta_serie, baud_rate, timeout=0.1)
    print(f"Conectado ao GPS na porta {porta_serie}\n")

    while True:
        linha = gps.readline().decode("utf-8",
errors="ignore").strip()

        if linha.startswith(("$GPGGA", "$GPRMC", "$GPGLL", "$GPGST")):
            resultado = parser.read_nmea(linha)
            if resultado:
                print(resultado)
                udp_socket.sendto((resultado + '\n').encode('utf-8'),
(UDP_IP, UDP_PORT))
            if parser.has_valid_fix():
                writer.write(parser.last_data)

except serial.SerialException:
    print(f"Erro: Não foi possível ligar à porta {porta_serie}.")
except KeyboardInterrupt:
    print("\nComunicação encerrada.")
```

```
finally:
    if 'gps' in locals() and gps.is_open:
        gps.close()
    udp_socket.close()
    print("[Main] Ligação UDP encerrada.")
```

Anexo B – parser.py

```
import pynmea2

class NMEAParser:
    def __init__(self):
        self.last_data = {
            "latitude": None,
            "longitude": None,
            "altitude": None,
            "hdop": None,
            "satellites": None,
            "speed": None,
            "timestamp": None,
            "rms": None,
            "std_dev_lat": None,
            "std_dev_long": None,
            "std_dev_alt": None
        }

    def read_nmea(self, sentence: str):
        try:
            msg = pynmea2.parse(sentence)
            tipo = msg.sentence_type

            if tipo == "GGA":
                if int(msg.gps_qual) == 0:
                    return "[GGA] Sem fix."

                self.last_data["latitude"] = msg.latitude
                self.last_data["longitude"] = msg.longitude
                self.last_data["altitude"] = msg.altitude
                self.last_data["hdop"] = float(msg.horizontal_dil) if
msg.horizontal_dil else None
                self.last_data["satellites"] = int(msg.num_sats) if
msg.num_sats else None

                return f"[GGA] Satélites: {msg.num_sats}, HDOP:
{msg.horizontal_dil}, Altitude: {msg.altitude} m"

            elif tipo == "RMC":
                self.last_data["latitude"] = msg.latitude
                self.last_data["longitude"] = msg.longitude
                self.last_data["speed"] = msg.spd_over_grnd
                self.last_data["timestamp"] = msg.datetime

                return f"[RMC] Hora: {msg.datetime}, Velocidade:
{msg.spd_over_grnd} nós"

            elif tipo == "GLL":
                self.last_data["latitude"] = msg.latitude
                self.last_data["longitude"] = msg.longitude
```

```
                return f"[GLL] Coordenadas: {msg.latitude}
{msg.lat_dir}, {msg.longitude} {msg.lon_dir}"

            elif msg.sentence_type == "GST":
    # Campos: hhmmss.sss,rms,maj,min,ori,std_lat,std_long,std_alt
                try:
                    self.last_data["rms"] = msg.rms
                except AttributeError:
                    self.last_data["rms"] = ""

                try:
                    self.last_data["std_dev_lat"] = msg.data[5] if
len(msg.data) > 5 else ""
                    self.last_data["std_dev_long"] = msg.data[6] if
len(msg.data) > 6 else ""
                    self.last_data["std_dev_alt"] = msg.data[7] if
len(msg.data) > 7 else ""
                except Exception:
                    self.last_data["std_dev_lat"] = ""
                    self.last_data["std_dev_long"] = ""
                    self.last_data["std_dev_alt"] = ""

                return f"[GST] RMS: {self.last_data['rms']}, " \
                    f"Erro Lat: {self.last_data['std_dev_lat']}, " \
                    f"Lon: {self.last_data['std_dev_long']}, " \
                    f"Alt: {self.last_data['std_dev_alt']}"

            else:
                return None

        except pynmea2.ParseError:
            return "[Erro] Mensagem inválida"

    def has_valid_fix(self):
        #verifica se tem fix valido
        return (
            self.last_data["latitude"] is not None and
            self.last_data["longitude"] is not None and
            self.last_data["altitude"] is not None and
            self.last_data["timestamp"] is not None
        )
```

Anexo C – xml_writer.py

```
import os
import xml.etree.ElementTree as ET
from datetime import datetime

class XMLWriter:
    def __init__(self, output_dir="output_xml"):
        os.makedirs(output_dir, exist_ok=True)
        self.output_dir = output_dir

    def write(self, data: dict):
        root = ET.Element("GPSData")

        for key, value in data.items():
            element = ET.SubElement(root, key)
            element.text = str(value) if value is not None else ""

        tree = ET.ElementTree(root)

        #timestamp no nome do ficheiro
        timestamp = data.get("timestamp")
        if isinstance(timestamp, datetime):
            filename = timestamp.strftime("gps_%Y%m%d_%H%M%S.xml")
        else:
            filename = "gps_unknown_timestamp.xml"

        filepath = os.path.join(self.output_dir, filename)
        tree.write(filepath, encoding="utf-8", xml_declaration=True)
        print(f"Ficheiro guardado: {filepath}")
```

Anexo D - socket_server.py

```
import socket
import struct

def start_udp_multicast_listener(multicast_group='224.0.0.1',
port=5000):
    sock = socket.socket(socket.AF_INET, socket.SOCK_DGRAM,
socket.IPPROTO_UDP)
    sock.setsockopt(socket.SOL_SOCKET, socket.SO_REUSEADDR, 1)
    sock.bind(('', port))

    mreq = struct.pack("4sl", socket.inet_aton(multicast_group),
socket.INADDR_ANY)
    sock.setsockopt(socket.IPPROTO_IP, socket.IP_ADD_MEMBERSHIP, mreq)

    print(f"[Multicast Server] À escuta em {multicast_group}:{port}")
    while True:
        data, addr = sock.recvfrom(4096)
        print(f"[Multicast Server] De {addr}: {data.decode('utf-8')}")

if __name__ == '__main__':
    start_udp_multicast_listener()
```

Anexo E – gui_viewer.py

```
import tkinter as tk
from tkinter import ttk
import socket
import struct
import threading

class XMLViewerApp:
    def __init__(self, root, multicast_group='224.0.0.1', port=5000):
        self.root = root
        self.root.title("Visualização de Dados GPS")

        self.tree = ttk.Treeview(root, columns=("Campo", "Valor"),
show='headings')
        self.tree.heading("Campo", text="Campo")
        self.tree.heading("Valor", text="Valor")
        self.tree.pack(fill=tk.BOTH, expand=True)

        threading.Thread(target=self.listen_multicast,
args=(multicast_group, port), daemon=True).start()

    def listen_multicast(self, group, port):
        sock = socket.socket(socket.AF_INET, socket.SOCK_DGRAM,
socket.IPPROTO_UDP)
        sock.setsockopt(socket.SOL_SOCKET, socket.SO_REUSEADDR, 1)
        sock.bind(('', port))

        mreq = struct.pack("4sl", socket.inet_aton(group),
socket.INADDR_ANY)
        sock.setsockopt(socket.IPPROTO_IP, socket.IP_ADD_MEMBERSHIP,
mreq)

        while True:
            data, _ = sock.recvfrom(4096)
            message = data.decode('utf-8')
            self.root.after(0, self.update_table, message)

    def update_table(self, message):
        parsed = {}
        for pair in message.strip().split(';'):
            if '=' in pair:
                key, val = pair.split('=', 1)
                parsed[key.strip()] = val.strip()

        self.tree.delete(*self.tree.get_children())
        for campo, valor in parsed.items():
            self.tree.insert("", "end", values=(campo, valor))
```

```
def run_viewer():
    root = tk.Tk()
    app = XMLViewerApp(root)
    root.mainloop()
```

Anexo F – deparser.py

```
import xml.etree.ElementTree as ET
from datetime import datetime

class NMEADeparser:
    def __init__(self, xml_file_path):
        self.xml_file_path = xml_file_path

    def read_xml(self):
        """Lê o XML e devolve um dicionário {campo: valor}"""
        try:
            tree = ET.parse(self.xml_file_path)
            root = tree.getroot()
            data = {child.tag: child.text for child in root}
            return data
        except Exception as e:
            print(f"[Deparser] Erro ao ler XML: {e}")
            return None

 #checksum

    def add_checksum(self, sentence):
        if not sentence.startswith("$"):
            sentence = "$" + sentence

        body = sentence[1:].split("*")[0]
        checksum = 0
        for c in body:
            checksum ^= ord(c)
        return f"{sentence.split('*')[0]}*{checksum:02X}"

# frases NMEA

    def to_gga(self, data):
        try:
            timestamp = datetime.fromisoformat(data['timestamp'])
            time_str = timestamp.strftime('%H%M%S')

            lat = self.convert_lat(data['latitude'])
            lat_dir = "N" if float(data['latitude']) >= 0 else "S"
            lon = self.convert_lon(data['longitude'])
            lon_dir = "E" if float(data['longitude']) >= 0 else "W"

            gga = (
                f"$GPGGA,{time_str},{lat},{lat_dir},{lon},{lon_dir},1,
"
                f"{data['satellites']},{data['hdop']},{data['altitude'
]},M,,,"
            )
            return self.add_checksum(gga)
        except Exception as e:
```

```
            print(f"[Deparser] Erro ao gerar GGA: {e}")
            return None

    def to_gll(self, data):
        try:
            timestamp = datetime.fromisoformat(data['timestamp'])
            time_str = timestamp.strftime('%H%M%S')

            lat = self.convert_lat(data['latitude'])
            lat_dir = "N" if float(data['latitude']) >= 0 else "S"
            lon = self.convert_lon(data['longitude'])
            lon_dir = "E" if float(data['longitude']) >= 0 else "W"

            gll =
f"$GPGLL,{lat},{lat_dir},{lon},{lon_dir},{time_str},A"
            return self.add_checksum(gll)
        except Exception as e:
            print(f"[Deparser] Erro ao gerar GLL: {e}")
            return None

    def to_rmc(self, data):
        try:
            timestamp = datetime.fromisoformat(data['timestamp'])
            time_str = timestamp.strftime('%H%M%S')
            date_str = timestamp.strftime('%d%m%y')

            lat = self.convert_lat(data['latitude'])
            lat_dir = "N" if float(data['latitude']) >= 0 else "S"
            lon = self.convert_lon(data['longitude'])
            lon_dir = "E" if float(data['longitude']) >= 0 else "W"

            speed = float(data.get('speed', 0)) * 1.94384  # m/s →
knots

            rmc = (
                f"$GPRMC,{time_str},A,{lat},{lat_dir},{lon},{lon_dir},
"
                f"{speed:.1f},0.0,{date_str},,,A"
            )
            return self.add_checksum(rmc)
        except Exception as e:
            print(f"[Deparser] Erro ao gerar RMC: {e}")
            return None

    def to_gst(self, data):
        try:
            timestamp = datetime.fromisoformat(data['timestamp'])
            time_str = timestamp.strftime('%H%M%S')

            gst = (
                f"$GPGST,{time_str},{data.get('rms','')},"
```

```
                f"{data.get('std_dev_lat','')},{data.get('std_dev_long
','')},0.0,"
                f"{data.get('std_dev_alt','')},0.0,0.0"
            )
            return self.add_checksum(gst)
        except Exception as e:
            print(f"[Deparser] Erro ao gerar GST: {e}")
            return None

#conversao extra

    def convert_lat(self, lat_str):
        lat = abs(float(lat_str))
        degrees = int(lat)
        minutes = (lat - degrees) * 60
        return f"{degrees:02d}{minutes:07.4f}"

    def convert_lon(self, lon_str):
        lon = abs(float(lon_str))
        degrees = int(lon)
        minutes = (lon - degrees) * 60
        return f"{degrees:03d}{minutes:07.4f}"
```

Anexo G - gps_stream.service

Para criar o ficheiro de serviço *gps_stream.service*, é necessário correr a seguinte linha de código na Linha de Comando:

```
sudo nano /etc/systemd/system/gps_stream.service
```

Aplicar o conteúdo seguinte ao ficheiro, tendo o cuidado de ajustar o caminho do ficheiro *main.py*.

```
[Unit]
Description=GPS NMEA to UDP Multicast
After=network.target

[Service]
ExecStart=/usr/bin/python3 /home/pi/nmeaprojeto/ficheiros/main.py
WorkingDirectory=/home/pi/nmeaprojeto/ficheiros
StandardOutput=inherit
StandardError=inherit
Restart=always
User=pi

[Install]
WantedBy=multi-user.target
```

Gravar e fechar o mesmo através dos seguintes comandos: Ctrl+O seguido de Enter e Ctrl+X.

Ativar o ficheiro de serviço correndo as linhas de código abaixo na Linha de Comando:

```
sudo systemctl daemon-reload
sudo systemctl enable gps_stream.service
sudo systemctl start gps_stream.service
```

Por fim, é necessário verificar se o ficheiro está efetivamente a correr através da seguinte linha de código:

```
systemctl status gps_stream.service
```

Anexo H– fake_gps_input.py

```
import time
from parser import NMEAParser
from xml_writer import XMLWriter

parser = NMEAParser()
xml_writer = XMLWriter()

fake_sentences = [
    # GGA
    "$GPGGA,123519,4807.038,N,01131.000,E,1,08,0.9,545.4,M,46.9,M,,*47
",
    # RMC
    "$GPRMC,123520,A,4807.038,N,01131.000,E,022.4,084.4,230394,003.1,W
*60",
    # GLL
    "$GPGLL,4916.45,N,12311.12,W,225444,A,*1D",
    # GST
    "$GPGST,123521,1.2,0.7,0.6,45.0,0.5,0.4,1.0*4C"
]

print("[FakeGPS] A transmitir mensagens falsas...\n")

try:
    while True:
        for sentence in fake_sentences:
            result = parser.read_nmea(sentence)
            if result:
                print(result)  # Mostra no terminal
                if parser.has_valid_fix():
                    # Guardar em XML
                    xml_writer.write(parser.last_data)

            time.sleep(1)  # intervalo entre mensagens

except KeyboardInterrupt:
    print("\n[FakeGPS] Emissão terminada pelo utilizador.")
```